uc3m | Universidad Carlos III de Madrid

Departamento de Ingeniería de Sistemas y Automática

Grado en Ingeniería Electrónica Industrial y Automática

2018 - 2019

Trabajo Fin de Grado

"Visión artificial aplicada en la identificación de objetos y su parametrización geométrica"

Autor: Carlos Justo de Frías

Tutor/es:

Juan Hernández Vicén

Santiago Martínez de la Casa Díaz

Leganés, Madrid

Agradecimientos

Sinceramente, este apartado, aunque lo haya dejado para el final, es un apartado que realmente tenía ganas de escribir, ya que me apetecía agradecer a todas esas personas que me han ayudado a ser lo que soy actualmente, y a que pueda terminar el Grado de Ingeniería Electrónica Industrial y Automática con este Trabajo Fin de Carrera.

Como no puede ser de otra forma, en primer lugar, me gustaría dar las gracias a mis padres, que son, con diferencia, los que más me han ayudado durante toda mi vida, tanto en los buenos momentos como en los malos. Mención especial a mi hermana, que me lleva aguantando y apoyando desde que el día que nací, labor que, seguramente, haya sido bastante complicada para ella.

También quiero agradecer al resto de mi familia por sus constante ayudas. apoyos e incluso, halagos.

No puedo olvidarme de dar las gracias a todos y cada uno de los amigos que he ido haciendo durante mi vida, ya sean del barrio, de la universidad, de mi pueblo, de actividades extraescolares o de donde sea. Cada uno de vosotros ha aportado su granito de arena para que yo hoy esté aquí.

Me gustaría hacer un agradecimiento especial a mi novia, la cual me ha tenido que aguantar durante todo el proyecto, cosa que no es fácil, apoyándome y confiando en que podía terminarlo.

Me gustaría agradecer también a todos los profesores que me han dado clase en el colegio o en la universidad. Mención especial a Don Pedro, que, aunque no pueda leerlo, es el profesor que más me ha marcado en mi vida y el que hizo que me gustaran las ciencias tanto como me gustan ahora.

Dentro del ámbito universitario, quiero agradecer a mi tutor, Juan Hernández, por haberme dado la oportunidad de realizar este proyecto y por sus constantes consejos y ayudas.

Finalmente, me gustaría agradecer a la comunidad científica que, de forma desinteresada, comparten sus conocimientos a través de internet.

Resumen

El principal objetivo de este proyecto es desarrollar un software basado en visión artificial para identificar y distinguir una caja en el entorno que rodea a un robot humanoide, obteniendo las características determinantes que la definen, tales como el perímetro en 2D, el centro de masas, o la posición relativa de la caja respecto a un origen de referencias en el robot. La información del entorno se consigue gracias a una cámara de visión con sensor de profundidad que se utiliza para obtener distancias del entorno.

Los experimentos llevados a cabo para el desarrollo del software han sido realizados en el robot humanoide TEO (Task Environment Operator), del grupo de investigación de la Universidad Carlos III, RoboticsLab.

Para obtener todos los datos necesarios y, con ellos, los objetivos del proyecto, se han realizado una serie de pasos secuenciales, aumentándose la complejidad de lo realizado en cada paso. En un primer momento se estudiaron las técnicas existentes de visión artificial sobre la identificación de objetos para estar más informado de los últimos avances y de esta forma, poder elegir qué camino tomar en el desarrollo. Una vez elegido el método a seguir, se realizó un filtrado y acondicionamiento de la imagen para poder identificar correctamente la caja. Finalmente se obtuvieron sus características geométricas, así como su posición relativa y los puntos finales donde el robot TEO tendría que manipularla.

Este software, con los datos obtenidos de él, será utilizado posteriormente en otro proyecto el cual tendrá como objetivo la manipulación de la caja, como cogerla o abrirla, con el robot humanoide del laboratorio, TEO. De esta forma este trabajo es la fase inicial de ese proyecto futuro. Así sabemos cómo está colocada la caja y a qué distancia del robot para que, posteriormente, se puedan realizar las operaciones necesarias para su manipulación.

Abstract

The main objective of this project is to develop a software based on artificial vision to identify and distinguish a box in the environment surrounding a humanoid robot, and to obtain the defining characteristics of the box, such as the perimeter in 2D, the center of masses or its relative position compared to a reference source in the robot. The information of the surrounding environment such as the distance, is obtained with a view camera with a depth sensor.

In order to develop the software, the experiments have been carried out in the humanoid robot TEO (Task Environment Operator) of the research group of the Universidad Carlos III, RoboticsLab.

A series of sequential steps have been carried out, increasing their complexity on each step, to obtain the necessary data and to fulfill the objectives of the project. At first, the existing techniques on the identification of objects on artificial vision were studied to be more informed about the latest developments and, in this way, being able to choose a path in the development. Once the method to be followed was selected, a filtering and a conditioning of the image was performed to identify correctly the box. Finally, its geometric characteristics were obtained, as well as its relative position and the final points that TEO, the robot, would use to manipulate the box.

This software, with the data obtained from it, will then be used in another project which will aim to manipulate the box, such as picking it up or opening it with the humanoid robot of the laboratory, TEO. In this way, this work is the initial phase of that future project. It's necessary to know where the box is placed to, later on, perform the necessary operations for its handling

Índice

Índice de ilustraciones

Índice de Gráficos

Índice de Tablas

Capítulo 1. Introducción

Se considera que los primeros robots surgen en el antiguo egipcio sobre el año 2000a.c., con las esculturas egipcias animadas. Sin embargo, a estos primeros robots se les pueden definir más bien como artilugios mecánicos con algún tipo de automatización que conseguía que se moviera sin labor humana.

Tras ello, surgieron otros tipos de mecanismos automatizados que podrían considerarse robots. No obstante, sus creadores no sabían que lo que estaban creando fuera un robot como tal, ni siquiera conocían la palabra robot, ya que no fue hasta 1920 cuando se usó por primera vez dicha palabra. Fue en la novela llamada RUR (Rossums Universal Robots) de Karel Capek donde se encuentra la primera aparición del término "robot", que proviene de la palabra checa "robota", la cual significa fuerza de trabajo haciendo referencia a siervos (Sánchez Martín, 2007).

Avanzaban los años y la robótica no acababa de despegar ni de tener importancia en el día a día, los únicos proyectos que se realizaban en la robótica eran acciones muy repetitivas donde el robot no procesaba nada y solo se encargaban de mover objetos de un punto a otro, como el primer robot comercial *Ultimate* de Unimation (Universal Automation), que podía transportar piezas fundidas en un molde hasta una cadena de montaje (Gómez, 2013).

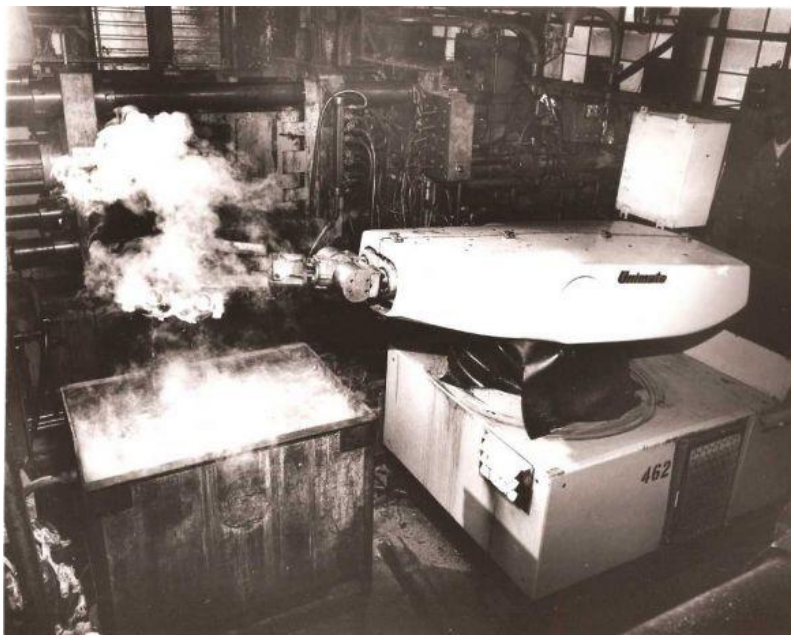

Ilustración 1 – Unimation, primer robot comercial

Sin embargo, esta situación cambió con el libro de Isaac Asimov “Yo Robot”, donde se citan las 3 leyes de la robótica. Este libro significo un cambio en la mentalidad de la robótica, dejando de lado los sistemas totalmente planificados para dar más importancia a los sistemas autónomos con capacidad de recibir información, procesarla y de actuar según la información que recibe. Y de esta necesidad de obtener información del entorno surgió la visión artificial, siendo actualmente la mejor forma para conocer qué nos rodea.

1.1. Motivación

En las últimas décadas el avance y posibilidades de la visión artificial ha aumentado enormemente. No solo en aplicaciones orientadas a la robótica de humanoides, sino también en diversos campos como la industria (en la que se controlan los parámetros ambientales que afectan la visión o para comprobar que no haya ningún obstáculo que impida la acción a realizar), la automoción con los vehículos autónomos, la seguridad o incluso en la sanidad.

De esta forma, tras haber cursado una asignatura de sistemas de percepción donde se estudiaron las bases de la visión artificial en 2D me propuse realizar un proyecto de fin de carrera que me permitiera conocer más este mundo añadiendo aspectos como un mejor filtrado de las características en una imagen en 2D, mezcla de información en 2D con profundidad o probar algoritmos de inteligencia artificial que usan redes neuronales.

Sin embargo, aunque la visión artificial fuera algo que me gustaba y de lo que quería obtener más conocimientos, lo que más me ha gustado y de lo que realmente me apetecía profundizar más era la programación visual (de prueba y error, como lo llamo yo), una programación que después de escribir un código obtienes resultados tangibles por lo que se puede comprobar de manera visual las consecuencias de lo que se ha programado y ver como se desempeña.

Con estas ideas en mente encontré el departamento de sistemas y automática donde había un robot humanoide (TEO) centrado en investigaciones y con una cámara de profundidad en la cabeza donde se realizaban tanto trabajos fin de carrera como de máster. En dicho departamento se cumplían los dos aspectos en los que quería centrarme, una programación con resultados tangibles utilizando la visión artificial.

De esta manera surgió mi proyecto de visión artificial orientada a la identificación y obtención de las características de un objeto colocado en el entorno del robot para que posteriormente el robot pudiera manipular dicho objeto.

1.2. Objetivos

El objetivo principal del proyecto es el desarrollo de un software basado en visión que sea capaz de identificar y detectar correctamente la posición de una caja en el entorno para que en trabajos futuros, se puedan calcular las trayectorias necesarias para manipular dicha caja.

Para facilitar el trabajo y poder alcanzar el objetivo principal del TFG, se ha dividido en diferentes subobjetivos:

- En primer lugar, trabajando con imágenes en 2D, habría que distinguir la caja correctamente del entorno (con una segmentación), obteniendo su perímetro y los puntos de las esquinas de las cajas.
- Posteriormente, se abandonarían las imágenes 2D para trabajar con profundidad. Una vez detectada correctamente la caja en el entorno, se obtendrían las distancias a la que están los puntos más característicos y las distancias entre los diferentes planos que forman las caras de la caja. También se comprobaría que dichos puntos tienen coherencia y se situarían en el espacio realizando los cálculos y operaciones necesarias, con 3 coordenadas, tomando como sistema de referencia un punto conocido del robot.
- Con todos los puntos ya obtenidos, se calcularían matemáticamente los puntos medios de las caras. Dichos puntos son lo que en un futuro el robot utilizará para realizar las operaciones de manipulación correspondientes.

1.3. Marco socioeconómico

La visión artificial y la robótica han abierto un sinfín de nuevas oportunidades en la sociedad en las últimas décadas. La era de la inteligencia artificial, aunque aún le queda un tiempo, cada vez está más cerca de ser una realidad, dejando atrás la época donde solo se veía en las obras de ciencia ficción.

Esta tecnología ha llegado a nuestra sociedad casi sin enterarnos, permitiendo realizar acciones complejas que hace unos años no se podían ni imaginar. Algunos ejemplos actuales de la visión artificial puede ser la conducción autónoma en los vehículos o drones, como utiliza Amazon para entregar paquetes en zonas difíciles de acceder por coche, o el reconocimiento y manipulación de objetos por los brazos industriales o los robots humanoides.

La visión artificial y la robótica actualmente tienen un gran papel social a través de los robots sociales. Impacto que con el paso de los años va a ir creciendo de manera exponencial con la normalización del uso de los robots y el abaratamiento en la producción de estos. Estos robots, utilizando la visión artificial, permiten interactuar con el ser humano o con otros robots y se centran en ayudar a los humanos ya sea porque necesitan algún servicio o para ofrecerles ayuda para poder vivir mejor en el día a día. Un ejemplo puede ser los robots que se encuentran en algunos hoteles en Japón, donde realizan la función de recepcionistas, llevando a las personas hasta su habitación, tomando nota de las reservas de la gente e incluso puede regular la temperatura de la habitación a través de comandos de voz (West, 2015).

Sin irnos tan lejos, en España ya hay varios robots utilizándose para ayudar en la rehabilitación de personas y para asistir a las personas dependientes. En la Unidad de Geriatría del Complejo Hospitalario de Cáceres y Hospital de Día se utiliza diariamente un robot desarrollado para interaccionar con pacientes con la enfermedad de Parkinson (Núñez, 2011).

Prácticamente en la mayoría de los sectores laborables se puede utilizar la visión artificial para mejorar la eficiencia e incluso para sustituir la mano de obra humana por una máquina. Las ventajas de este cambio son numerosas, por ejemplo, las máquinas no se ponen enferma ni piden vacaciones, y si hay que arreglar o modificar el robot no lleva tanto tiempo como el tiempo de aprendizaje de una persona o la búsqueda de nuevo personal. Además, esta tecnología puede trabajar las 24 horas al día y sin desconcentrarse debido a algún agente externo.

No obstante, la inclusión de la visión artificial y la robótica en la sociedad tiene una parte negativa, la reducción de puestos de trabajo en donde esta tecnología se empieza a incluir. El 100% de los puestos no se sustituyen por las nuevas tecnologías ya que debe haber personal que coordine, apoye y supervisen sus labores, pero si podría provocar que en un futuro mucha gente fuera despedida o recolocadas en otros puestos. Sin embargo, esto no sería de golpe, sino que primero veremos a robots trabajando juntos a otros humanos en vez de reemplazándolos (Rotman, 2013).

La gran consultora internacional PwC estima que para el 2030 el 34% de los trabajos podrían estar automatizados. Otra consultora, Accenture, determinó en 2017 que 17000 puestos de trabajo actuales de su empresa, en un futuro cercano iban a ser ocupados por maquinas. A pesar de ello, Accenture afirmó que todos los trabajadores que perderían sus puestos debido a las máquinas serían recolocados en otro puesto dentro de la empresa, evitando perder el trabajo (Berriman, 2017; Cappella, 2017).

Es importante aclarar que, aunque cada día haya más empleos que puedan realizar robots, no significa que el número de empleos para los humanos disminuya. Cada poco tiempo surgen nuevos trabajos relacionados con la visión artificial y la robótica que necesitan ser realizados por seres humanos.

Por esto último, la capacidad de adaptación de la sociedad y la educación para adaptarse a esta nueva forma de vida donde los robots estarán presentes en nuestros día a día es fundamental. Si la educación y mentalidad de todos los países no consigue adaptarse tan rápido como avanza la investigación y desarrollo de los robots, es probable que en el futuro puedan existir problemas de trabajo o rechazo a estas nuevas tecnologías.

Para finalizar, la inclusión de la robótica en nuestra sociedad es realmente útil e importante que se realice debido a que mejora la calidad de vida. Es una tecnología que va a mejorar enormemente la sociedad donde vivimos, pero debe de ir acompañada de una evolución en la educación y mentalidad de los humanos para que no existan posibles problemas en el futuro.

1.4. Marco regulador

Actualmente no existe ningún tipo de normativa que regule el uso de la visión artificial en nuestro país. Es cierto que existe la ley de Protección de Datos de Carácter Personal (Ley Orgánica 15/1999, de 13 de diciembre) pero en el caso del proyecto no se aplica debido al uso de la visión artificial de manera no invasiva. Los objetos a identificar son cajas, no se busca en ningún momento información sobre personas.

Además, el proyecto se realiza sobre el robot humanoide de la universidad de Carlos III, TEO, el cual solo tiene fines de educación e investigación para los propios estudiantes de la universidad, sin fines comerciales, por lo que no se aplicaría ninguna regulación al realizarse todas las pruebas dentro del laboratorio.

Sin embargo, es obvio que cuanto mayor sea la inteligencia y visión artificial en los robots, mayor será su autonomía y en consecuencia tendrán menos dependencia de los usuarios. Por eso, es un hecho que la legislación debe adaptarse para los posibles problemas que surgirán y que las normas actuales no dan respuesta (González, 2017).

El primer paso para la realización de la regulación sobre la robótica a nivel europeo ha sido la elaboración el 31 de mayo de 2016, de un informe que recoge distintas recomendaciones sobre normas de Derecho civil acerca de la robótica con el fin de "asegurar que los robots estén y sigan estando al servicio de los seres humanos" (González, 2017).

Entre las recomendaciones incluidas en el informe se destacan las siguientes:

- Reglas de responsabilidad por los daños causados por los robots.
- Creación de un estatuto de persona electrónica.
- Creación de una *Agencia Europea de Robótica e inteligencia artificial* que ayude a las autoridades públicas asesorando con sus conocimientos
- Creación de un Registro Europeo de robots autónomos.

Todas estas recomendaciones están realizadas con el fin de que la regulación siga construyendo una sociedad centrada en los seres humanos y no en los robots, los cuales carecen de sentimientos y consciencia. Deben establecer unos límites en el diseño y aplicaciones para que la seguridad de las personas no se vea afectadas por la inclusión de la robótica.

1.5. Descripción de los capítulos

A continuación, se detalla la forma en la que se ha estructurado la memoria y un breve resumen del contenido de cada capítulo.

- Capítulo 1: Introducción al proyecto, presentando los motivos que llevaron a la elección y realización de este, así como los objetivos que se marcaron al principio y un análisis sobre el posible impacto del proyecto en la sociedad y las leyes que actualmente se podrían aplicar.
- Capítulo 2: Se hará un breve repaso de cómo y dónde surgió la visión artificial, sus usos actuales y diferentes métodos utilizados actualmente, haciendo hincapié en la detección de objetos. También se hará un breve repaso de las cámaras utilizadas para obtener imágenes a color con profundidad.
- Capítulo 3: En este apartado se nombrarán todos los recursos, tanto de hardware como de software, utilizados para la realización del proyecto y se detallará el presupuesto realizado.
- Capítulo 4: Explicación detallada de la metodología usada para la realización del proyecto, así como todas las acciones realizadas cronológicamente y ejemplos gráficos de su aplicación en el proyecto.
- Capítulo 5: En este apartado se nombrarán diferentes desarrollos extra que se han realizado que no estaban incluidos en el proyecto inicial pero que serán de bastante utilidad para el objetivo final.
- Capítulo 6: Exposición de los resultados obtenidos y con comentarios de dichos resultados.
- Capítulo 7: Conclusiones obtenidas a partir del desarrollo y trabajos futuros que se pueden realizar en el entorno del proyecto.
- Capítulo 8: Bibliografía utilizada en el proyecto.

Capítulo 2. Estado del Arte

En este capítulo se va a tratar el estado del arte de la visión por computación, así como el reconocimiento de objetos, comparando diferentes métodos existentes, y de las cámaras de visión, utilizadas para obtener la imagen y la profundidad. Para ello, se hará un resumen de cómo ha ido evolucionando las tecnologías en su historia hasta el día de hoy, dando detalles al final del estado actual de ellas. También se hablará de diferentes alternativas que surgieron o que actualmente existen.

2.1. Visión por computador

En la Antigua Grecia, Aristóteles afirmó que "la visión es saber qué hay y dónde mediante la vista", lo cual es correcto. Sin embargo, la definición que mejor describe el concepto de visión por computación es la que formuló Gibson (1979): "visión es recuperar de la información de los sentidos (vista) propiedades validas del mundo exterior".

El ser humano consigue identificar los objetos del entorno y su posición procesando la información que le llega al cerebro obtenida a través de nuestros ojos, siendo estas imprescindible para las actividades del día a día de una persona. La visión artificial, o visión computacional, pretende replicar esta capacidad en los ordenadores, de forma que se puedan detectar diferentes objetos en el espacio y su posición a través de la imagen que se recibe, por ejemplo, con una cámara. Pero, ¿cómo funciona realmente la visión humana? (Sucar & Gómez, 2011).

2.1.1. Visión humana

Los seres humanos percibimos las imágenes del mundo exterior a través del ojo, concretamente a través de la retina, que es por donde la luz es capaz de entrar para posteriormente, con esa luz, obtener la información del entorno. La visión humana funciona con un sistema de visión binocular y estereoscópico debido a que implica la intervención simultánea

de los dos ojos obteniendo dos imágenes de un objeto, que, al fundirse en una, producen una sensación de relieve por estar tomadas con un ángulo diferente para cada ojo (Sanabria S. & Archila D., 2011).

Para realizar esta acción, la retina posee dos tipos de células que actúan como minúsculos receptores: los bastones y los conos. Los bastones son sensibles a la intensidad lumínica mientras que los conos son los encargados de detectar el color. Hay tres tipos de células conos, que se diferencian porque cada una puede recibir la intensidad de un único color: el rojo, el verde y el azul. Gracias a este descubrimiento se estableció la teoría tricromática de la percepción del color, según la cual se indica que todo color es una mezcla de estos tres colores primarios. De este descubrimiento también surgió el campo de visión RGB tan utilizado en la visión por ordenador, entre otros campos (Espasa, 2004).

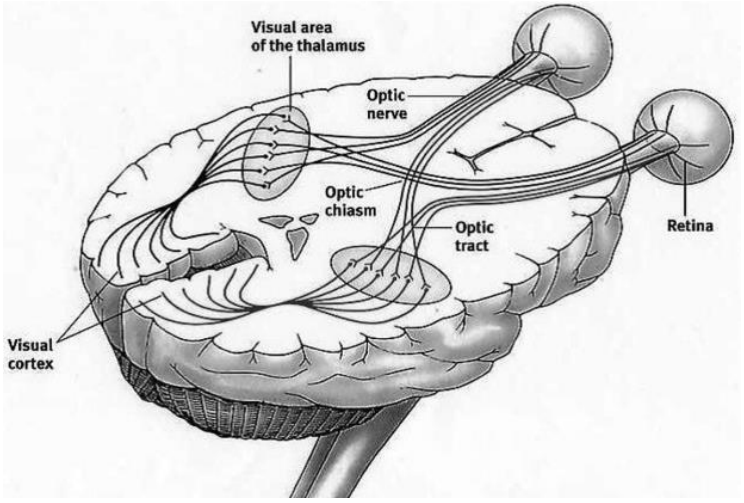

Ilustración 2 – Sistema de visión en los seres humanos

Una vez que los receptores del ojo han recibido toda la información del entorno, se transforma en impulsos nerviosos y se envía al cerebro a través del nervio óptico donde será procesada en el córtex visual (Alberich, Gomez F., & Ferrer F, 2011).

En el córtex visual la información es analizada por capas de neuronas altamente especializadas, teniendo cada capa diferentes funciones. La primera capa se especializa en detectar los bordes de los objetos en las imágenes, los límites, donde existe una diferencia de contraste o iluminación muy grande. Las demás capas son las encargadas de detectar contornos o características de la imagen como puede ser la profundidad, el movimiento o el color (Alberich, Gomez F., & Ferrer F, 2011).

La visión artificial se basa principalmente en este último concepto de capas. No se realizan acciones muy complejas que intenten detectar toda la información del objeto a detectar

a la primera, sino que se intenta ir sacando la información del objeto poco a poco, debido a que actualmente no es muy accesible una tecnología capaz de realizar todo ese procesamiento (Ullman, 1996). Pero esta idea de visión artificial se fue modificando a lo largo de la historia hasta llegar a la actualidad.

2.1.2. Historia de la visión artificial

Diferentes autores han identificado la historia de la visión artificial como elemento de partida a la hora de crear sus teorías. En este apartado se seguirá, principalmente, la división cronológica indicada por Arturo de la Escalera (2001).

Las primeras investigaciones relacionadas con la visión artificial surgieron en la década de los años 50. La facilidad con la que los seres humanos podemos conocer todo lo que contiene nuestro entorno hizo creer que un ordenador podría interpretar las imágenes que recibe del exterior de una manera relativamente fácil (Sucar & Gómez, 2011).

Con esta idea, se consiguieron buenos resultados en algunas investigaciones, como una de Roberts (1963) en la que creó un programa que consiguió demostrar la posibilidad de obtener la descripción matemática de la posición y orientación de los objetos de una escena y reproducirla desde otra perspectiva, confirmando que lo que se había procesado a través de la cámara era correcto; o la de Wichman (1967) que con un equipo de una cámara de televisión conectada a un ordenador pudo identificar las posiciones de algunos objetos al momento. Sin embargo, en ambos casos se trataban de experimentos en un entorno muy controlado y con fuertes restricciones. Se basaban en proyectos simples, con imágenes binarias que procesaban conjuntos de píxeles sueltos (Shun, 2015; Mendieta, 2003).

A pesar de estos resultados y del entusiasmo inicial, la visión artificial no resultó tan fácil como se esperaba, y se acabó demostrando que es un problema muy complejo. En los años siguientes lo único que se consiguieron en las investigaciones fueron fracasos y frustraciones por no obtener buenos resultados. Esto se debía principalmente a que no existía una tecnología tan avanzada como para poder realizar procesamientos mejores. Sin embargo, en estos años se desarrollaron algoritmos muy importantes como los detectores de bordes de Roberts (1965), Prewitt (1970) o Sobel (1970) que se siguen utilizando hoy en día. Por aquel entonces, estos algoritmos eran aplicables solo para unos casos muy concretos, estaban muy limitados. Por todo esto, en los años 70 se puede apreciar una disminución en las investigaciones y desarrollos sobre

la visión artificial, fallando las predicciones que se hicieron a mediados de siglo y comenzando lo que se conoce como el "Invierno de la Inteligencia Artificial" (Ferro, 2018).

Toda esta dejación en la visión artificial se debió principalmente a que, al intentar entender objetos tridimensionales en una proyección 2D, mucha información se pierde, pudiendo haber infinidad de posibles soluciones. Otro motivo del abandono fueron las altas expectativas del principio debido a que en un ser humano todo el proceso de obtener las características del entorno se hace de manera inconsciente en el cerebro, mientras que en los ordenadores no es posible, ejecutando acciones que los humanos no realizamos (Ullman, 1996).

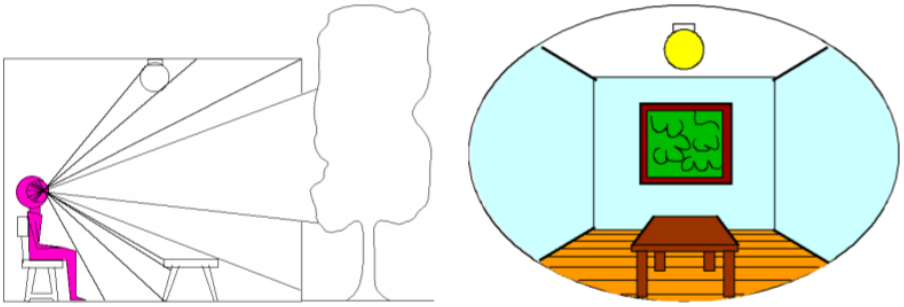

Ilustración 3 – Proyección 2D de una vista en 3D

Después de esta época de pesimismo, en la década de los 80 vuelven a aparecer algunas investigaciones, principalmente sobre extracción de características, como las de detección de texturas de Haralik y Shanmugam (1973), la detección de movimiento de Horn y Shunck (1981) o la interpretación de líneas de Lucas y Kanade (1981). Pero es a partir de esta década cuando la visión artificial vuelve a ser una de las principales líneas de investigación en el mundo con apariciones continuas en revistas científicas, mayores fondos para desarrollo e investigación y un mayor número de congresos. A su vez, esto fue posible debido a la mejora de procesamiento por parte de los ordenadores al ser cada vez más potentes y sofisticados e incluso con hardware específico para el procesamiento y tratamiento de imágenes.

2.1.3. Usos actuales de la visión por computador

Los numerosos avances en la visión artificial la han convertido en una herramienta muy utilizada en muchos campos hoy en día. Permiten la grabación de información, analizarla en el momento o guardar dichos datos para poder utilizarlos para un análisis o revisión posterior (Sanabria S. & Archila D., 2011; Vázquez, 2015).

Entre los campos más utilizados cabe destacar los siguientes:

Medicina

Dentro de la medicina hay varias ramas de investigación que introducen la visión artificial, algunas intentan "curar" la ceguera convirtiendo lo que se ve a través de una cámara en impulsos eléctricos que se enviarían al córtex visual, permitiendo así la visión a personas que no pueden ver (Dobelle, Quest, Antunes, Roberts, & Girvin, 1979).

Luego hay otras donde la visión artificial ayuda a los especialistas acondicionando las imágenes para una mejor interpretación, extrayendo la información necesaria o que detecta automáticamente enfermedades sin ayuda de intervención humana. Uno de los campos donde la visión artificial más se utiliza es en el análisis del sistema motor humano. (Villa Moreno, Gutiérrez Gutiérrez, & Pérez Moreno, 2008; Cho, Chao, Lin, & Chen, 2009)

Vigilancia/Seguridad

El aumento de las cámaras de seguridad en nuestro día a día para la vigilancia o seguridad de edificios, calles o zonas concretas hace que sea imposible que todas ellas sean controladas por operarios. De aquí nace la necesidad de utilizar la visión artificial en la seguridad y vigilancia.

Entre las diferentes funciones que puede realizar la visión artificial en la seguridad se puede destacar las de detección y seguimiento de una persona, identificación de personas o coches en un acceso y ver si dicha persona o coche puede ingresar o descubrir actividades o situaciones peligrosas. (Draghici, 1997; Cañas, Valencia, Restrepo, & Holguín, 2007; Lam, Cheung, & Liu, 2011)

Industria

Es el campo donde la visión artificial es más usada hoy en día. Tradicionalmente se utilizaba la intervención humana para la realización de este tipo de trabajo, pero el desarrollo tecnológico y la posibilidad de tener un entorno con la iluminación, las sombras y las piezas controladas, permite que se sustituyeran por máquinas capaces de ver, ofreciendo como mínimo la misma calidad a un precio menor. (Villán, y otros, 2011)

Aunque tiene una gran cantidad de aplicaciones posibles, donde más se utiliza es en el control de calidad, siendo la iluminación la parte más crítica ya que métodos diferentes de iluminación pueden tener efectos visuales muy diferentes. (Herrero, 2005)

Según el tipo de luz utilizada se puede extraer con mayor facilidad los datos que se buscan, como grietas, bordes, cambios de altura, entre otros. Para obtener esta información se puede utilizar tipos de iluminación como campo oscuro, iluminación coaxial, láseres o profundidad. (Secretaría de Estado de Educación y Formación profesional, 2012; Druzella, 2010).

Ilustración 4 – Luz ambiental vs campo oscuro

Aunque se obtienen muy buenos resultados, son resultados que se consiguen debido a que la luz está controlada. Las cámaras, de momento, son mucho menos sensibles que la visión humana y por eso las condiciones de iluminación deben de estar optimizadas al máximo. Este inconveniente es uno de los problemas que debe solucionar la visión artificial en el futuro para seguir avanzando.

Automoción

La visión artificial, junto a otros tipos de sensores, es imprescindible para la conducción autónoma, la cual se está llevando a cabo ya actualmente y que se seguirá utilizando en el futuro.

Se utiliza principalmente para la detección de carriles, la detección de objetos y para la detección de seres humanos. De esta forma se actuaría según lo que detecta del entorno el coche (Bertozzi M. , y otros, 2002; Broggi, Cerri, & Antonello, 2004).

Sin embargo, debido a las dificultades que tiene la conducción, la visión artificial en los vehículos aún no se ha desarrollado lo suficiente por lo que debe seguir mejorando con el futuro hasta conseguir vehículos 100% autónomos que no necesiten de acciones humanas para conducir (Bertozzi, Broggi, & Fascioli, 2000).

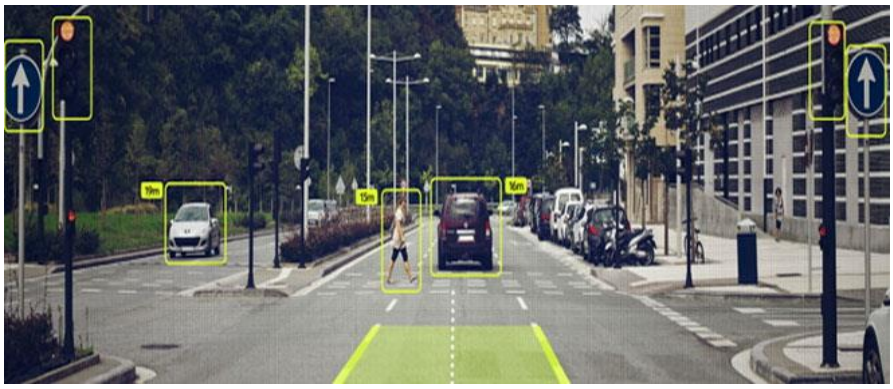

Ilustración 5 – Visión Artificial en vehículos

Aun así, actualmente la visión artificial no es el único sistema de percepción en los vehículos, funcionando a la vez que otros sensores como ultrasonidos o láser (fusión sensorial). Esto es necesario porque un vehículo que conduce a tales velocidades y que puede poner en peligro las vidas tanto dentro como fuera del vehículo, no puede únicamente fiarse de un solo tipo de sensores (Cox & Wilfong, 2012).

Robótica

Antiguamente los robots eran meros manipuladores, los cuales solo realizaban acciones rápidas, precisas y repetitivas. Eran sistemas totalmente planificados (de forma previa), incapaces de tomar ninguna decisión. Pero se llegó a una época donde la gente empezó a diseñar robots que hicieran más acciones de manera autónoma, necesitando obtener información del entorno para utilizarla y poder actuar consecuentemente en base a estos datos obtenidos (Aguilar & Angulo, 2012).

Para dar a los robots una autonomía real y poder realizar dichas acciones había que proveer a los robots con herramientas y sistemas de percepción y procesamiento. Además de los sensores, la percepción visual tiene un papel privilegiado en el análisis del mundo exterior siendo el que mejor se ha desempeñado en esa tarea ya que permite obtener más características del entorno (Ayache, 1991).

Las aplicaciones de visión artificial, así como los sistemas de visión por ordenador tienen funciones diferentes dependiendo del tipo de plataforma, o robot, en el cual se encuentran colocados. Como consecuencia de ello, existen multitud de aplicaciones y sistemas para cada tipo de función (Aguilar & Angulo, 2012).

Sin embargo, la visión artificial en los robots depende mucho de las condiciones lumínicas para funcionar correctamente. Los resultados pueden cambiar mucho dependiendo de la hora del día a la que se encuentre, y más con los robots móviles que se van desplazando en el exterior. Este problema es menor en los demás campos debido a que las labores se realizan en espacios controlados, menos en el de conducción autónoma, donde existe el mismo inconveniente que con los robots móviles.

Aun así, es un problema que ya está en vías de solucionarse y no evita que la visión artificial en la robótica no pueda utilizarse en alguna aplicación actualmente. Las más comunes son las encargadas de detectar obstáculos para poder después esquivarlos y seguir con una ruta, la de detectar humanos para poder interactuar con ellas, ya sea hablando o pausando una acción para que no exista peligro alguno, o la de identificación de objetos para posteriormente poder realizar alguna acción de manipulación sobre dicho objeto.

Sobre esta última aplicación, la de identificar objetos y la obtención de sus características, se profundizará más en el siguiente apartado debido a que es el tipo de aplicación que se ha utilizado y desarrollado en esta investigación.

2.2. Reconocimiento de objetos

Los seres humanos, así como la mayoría de las especies vivas del planeta Tierra, somos capaces de reconocer objetos en la vida real o en imágenes relativamente fácil en un periodo de tiempo y procesamiento cerebral relativamente corto. No importa como estén colocados dichos objetos, ni el tamaño ni la escala o si el objeto está obstruido por alguna vista. Esta labor que los seres humanos realizamos de forma tan fácil y sin pensar es una acción que a los sistemas de visión por ordenador se les ha complicado bastante desde el inicio de la visión artificial (Padilla, 2016).

Al igual que con la visión artificial, ocurre lo mismo con el tacto. Si a un ser humano se le coloca en la mano un llavero, seguramente desde el principio ya sabe qué es, y si no podría identificar bastantes características del objeto. Por otro lado, si a un robot humanoide le colocas en su mano un llavero lo único que detectará sería una variación del momento en algunas de sus articulaciones.

En la visión artificial, lo único que se recibe es una matriz de puntos (la equivalencia al ser humano es como si estuviéramos recibiendo esas señales eléctricas del ojo, pero el cerebro

no supiera que hacer con ellas o a que corresponden). Y en dicha matriz se tienen que realizar las operaciones necesarias para poder detectar en ellas los objetos.

Habitualmente, las plataformas elegidas para realizar esas operaciones son las basadas en programas secuenciales, sin embargo, han surgido en las últimas décadas otros métodos que "facilitaban" el trabajo, son las basadas en inteligencia artificial. Computacionalmente estas últimas técnicas son muy costosas de realizar, necesitando una gran cantidad de procesamiento, pero los resultados obtenidos son mejores que con los algoritmos secuenciales.

A continuación, se explicarán por separado los dos métodos utilizados para la detección de objetos que actualmente se utilizan además de las diferentes variantes que pueden tener cada uno.

2.2.1. Algoritmos secuenciales

Este tipo de operaciones se basan en el principio de que es imposible obtener de primeras toda la información que se busca en una imagen sin antes acondicionándola. Se realizan operaciones primarias previas para después, ya con las imágenes en las mejores condiciones, extraer las características finales.

De esta forma, tal y como dicen Ratha (1996) y Hamid (1994), la mayoría de los sistemas de visión actuales se pueden dividir en tres niveles diferentes, como se puede apreciar en la siguiente imagen.

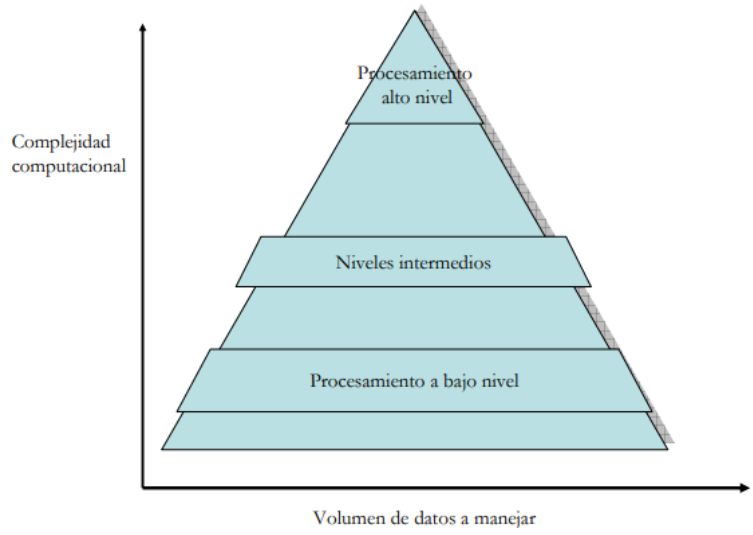

Ilustración 6 – Pirámide de volumen de datos vs complejidad computacional

Estos tres niveles se denominan como procesamiento de bajo nivel, procesamiento de nivel medio y procesamiento de alto nivel, realizándose en cada nivel operaciones diferentes.

- En el nivel inferior, el de procesamiento de bajo nivel, se realizan las operaciones donde se tienen en cuenta todos los píxeles de la imagen, utilizando operaciones sencillas como multiplicaciones, restas o sumas. Entre las operaciones que se suelen usar en este nivel se encuentran detección de bordes o filtrado de imágenes para acondicionarla y que sea más fácil extraer los datos de ella.
- El procesamiento de nivel medio se diferencia del nivel inferior, el de nivel bajo, por no utilizar todos los pixeles de la imagen. Son operaciones más complejas pero que solo se realizan en bloques o grupos de píxeles. Entre las operaciones que se pueden realizar en este nivel se encuentran a las operaciones de segmentación o las de etiquetado de zonas en diferentes regiones.
- Finalmente, en el procesamiento de alto nivel, encontramos operaciones y algoritmos que cuentan con una alta complejidad, siendo costosas de realizar. Algunos ejemplos de operaciones que se realizan en este nivel pueden ser operaciones de matching, de correlación o incluso si se cuenta con una cámara capaz de obtener profundidades, calcular la distancia a la que se encuentran varios puntos.

En la siguiente tabla se aprecia el resumen de los tres niveles que tienen los softwares que utilizan un formato secuencial.

Nivel	*Características*	*Ejemplos*
Bajo	Operaciones simples que utilizan todos los pixeles de la imagen. Manejo de gran cantidad de datos	Operaciones morfológicas simples, filtrado, modificación del histograma, detección de bordes
Medio	Operaciones más complejas que utiliza grupos o bloques de píxeles.	Conectividad entre píxeles, transformada de Hough para detectar líneas
Alto	Operaciones complejas que utiliza píxeles concretos de la imagen	Matching, correlación u obtención de datos en 3 dimensiones

Tabla 1 - Características y ejemplos de los diferentes niveles de operaciones secuenciales

Estos tres niveles resumen perfectamente la forma de actuación de la mayoría de los programas que utilizan visión artificial. No se suele trabajar con la imagen entrante sin antes acondicionarla de alguna forma para que resalte las características que se necesitan. Posteriormente se reduce el campo de trabajo reduciendo de esta forma los datos utilizados. Finalmente, se busca en ese campo reducido características más concretas para en el último paso extraer los datos necesarios de la imagen.

2.2.2. Inteligencia artificial

El reconocimiento de objetos pegó un gran salto cualitativo hace aproximadamente un lustro cuando el Machine y Deep Learning se empezaron a utilizar diariamente. Estas tecnologías están basadas en inteligencia artificial debido a que es el mismo programa el que va aprendiendo por sí mismo.

A diferencia del método anterior, donde todas las acciones que se realizaban eran para identificar uno o varios objetos muy concretos, en este método se puede conseguir un clasificador capaz de identificar varios objetos con diferencias entre ellos.

Sin embargo, la inteligencia artificial no solo está presente en los nuevos métodos de Machine y Deep Learning, sino que también se puede utilizar para resolver los problemas o las variaciones de los estados iniciales de los algoritmos secuenciales, acondicionando todas las posibles entradas a un mismo formato con el que trabajar. Esto es debido a que las imágenes digitales presentan mucha información difícil de analizar por una máquina y por eso se utilizan la inteligencia artificial para mejorar la interpretación. (Ramírez & Chacón, 2011).

Aun así, donde más está presente la inteligencia artificial es en los otros dos métodos ya nombrados.

Machine Learning

Los términos de Machine Learning e Inteligencia Artificial han cobrado gran popularidad en los últimos años, estando muy presente en nuestro día a día, aunque nosotros no lo sepamos. Sin embargo, ¿Cuál fue el origen de esta tecnología?

Mucha gente data el origen del Machine Learning, así como de la Inteligencia Artificial, con la creación del *Turing Test* de Alan Turing (1950), donde se lanzó la pregunta de que si las máquinas podían pensar. La prueba intentaba determinar si una máquina era realmente inteligente. Para pasar dicha prueba la máquina debería ser capaz de engañar a un humano haciéndole pensar que era una humana con quien hablaba en vez de una máquina.

Mientras parte de la comunidad científica considera el artículo de Turing como inicio de la inteligencia artificial y el test de Turing como la meta final, otra parte consideran que el Turing Test es inútil y que el origen data de 2 años más tarde, cuando Arthur Samuel escribe un programa capaz de aprender a jugar a las damas y a mejorar su juego con cada partida. (Saygin, Ciceklim, & Akman, 2000) (Buchanan, 2005).

Sin embargo, aunque hubo unas cuantas investigaciones en los años 60, en los años 70 dejaría de financiarse los proyectos sobre inteligencia artificial y no sería hasta el principio del siglo XXI cuando se volvería a investigar con fuerza todo lo relacionado con Machine Learning e Inteligencia artificial.

Esta explosión en el interés por el Machine Learning (y por la inteligencia artificial) se debe principalmente a dos razones: el aumento de la potencia de cálculo en los ordenadores y la gran abundancia de datos disponibles en la red en forma de bases de datos.

Una vez explicado el origen, hay que entender exactamente qué es, cómo funciona y como se aplican estas tecnologías al reconocimiento de objetos.

El Machine Learning, o aprendizaje automático o automatizado, tiene como objetivo desarrollar técnicas que permitan que los ordenadores, o máquinas con capacidad de procesamiento, aprendan. Pero las máquinas no aprenden por sí mismas, están programadas para adaptar un algoritmo conforme reciben datos. Cuantos más datos reciben, mejores serán los algoritmos que crean, siempre y cuando los datos introducidos sean datos fiables y de calidad, ya que si se mete algún dato erróneo puede resultar en un mal aprendizaje.

Esto es posible gracias a las redes neuronales que existen en el software de la máquina. Estas redes neuronales artificiales están inspiradas en el funcionamiento biológico de nuestro cerebro y las podemos definir como un conjunto de unidades de procesamiento que están interconectadas entre sí con la finalidad de recibir una señal de entrada, procesarla y emitir señales de salida (Huerta, Vásquez, Dueñas, Loayza, & Naupari, 2009).

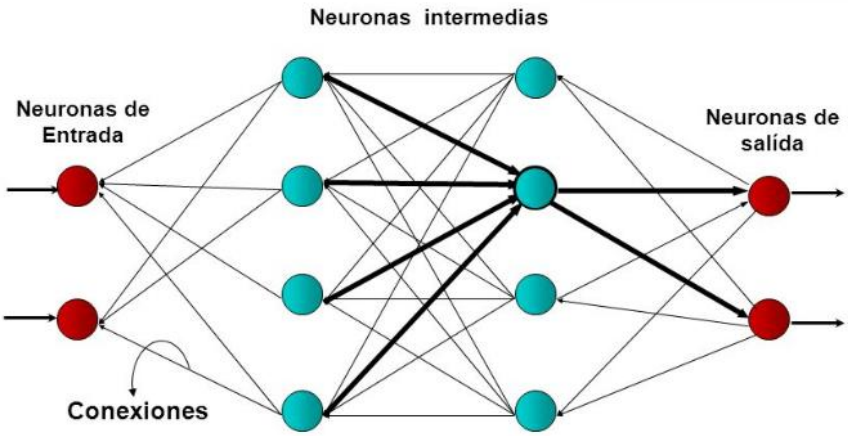

Ilustración 7 – Esquema de las Redes Neuronales Artificiales

Cada conexión tiene un valor, un peso, que se multiplica por los valores que le llegan y se transmite a las demás capas. A estas operaciones, desde que entra un dato, hasta que sale, se le llama función de activación. La salida se analiza para ver si se ha obtenido lo esperado y en caso negativo los valores de los pesos de las neuronas se van actualizando de manera iterativa buscando reducir el error. Este último proceso se llama *back-propagation*, debido a que los pesos se modifican desde los últimos a los primeros, y es la manera en la que se realiza el aprendizaje automático.

Por lo tanto, para reconocer objetos lo único que habría que hacer es introducir en nuestro código una gran cantidad de datos de los objetos que tiene que identificar, que aprenda y que posteriormente cree un algoritmo (un clasificador) que sea capaz de detectar los objetos en el entorno creando un cuadrado en la zona de la imagen que contenga el objeto (Alexe, Desealaers, & Ferrari, 2010).

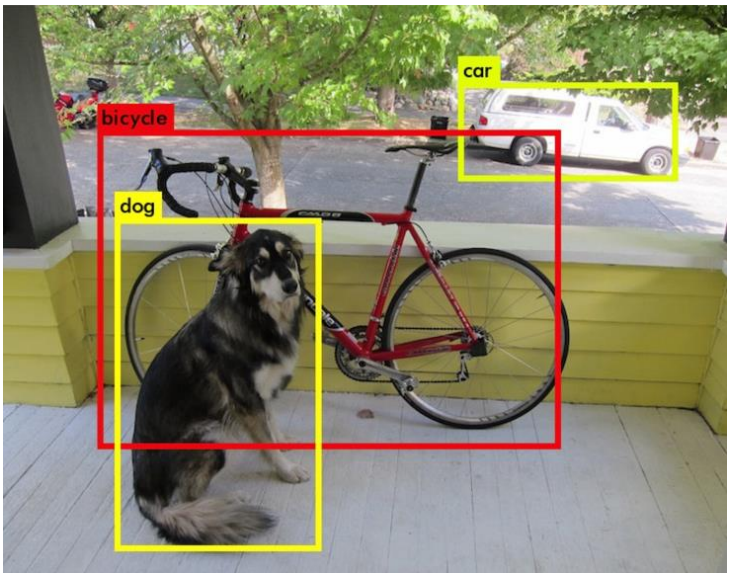

Ilustración 8 – Ejemplo de detección de objetos con Machine Learning

Deep Learning

Mientras que con el Machine Learning había que ayudar al ordenador indicándole qué eran los datos introducidos (o se conseguía un clasificador de si/no), en el Deep Learning se avanza en el concepto de aprendizaje no supervisado.

Deep Learning es una forma, o evolución, del Machine Learning que permite al ordenador aprender desde la experiencia y entender el mundo que percibe. Los algoritmos son capaces de aprender sin intervención humana previa, sacando ellos mismos las conclusiones de todos los datos (Goodfellow, Bengio, & Courville, 2016).

Este nuevo sistema es capaz no solo de aprender por sí mismo a reconocer imágenes, sino que también es capaz de dotar a los objetos que ve en ella de un significado y manejar conceptos abstractos, no solo cosas materiales. Son capaces de entender lo que están viendo.

Al tener como base al Machine Learning, las redes de neuronas automáticas siguen presentes en el software, realizando las mismas funciones, pero con un número de neuronas y capas bastante mayor y también más sofisticadas (Schmidhuber, 2015).

De esta forma, con Deep Learning reconocer objetos a tiempo real en un futuro cercano es posible. Incluso actualmente, muchas empresas ya están apostando por esta tecnología como Google, IBM o Amazon.

Un ejemplo de una aplicación actual de Deep Learning la encontramos en la conducción autónoma de Tesla o Uber, reconociendo ciertos patrones de conducción o de los objetos que aparecen frente al coche.

Otro ejemplo es un algoritmo desarrollado por investigadores de la universidad de Granada, que permite detectar en un video automáticamente a tiempo real la presencia de pistolas. La eficacia del algoritmo se ha probado tanto en videos de baja calidad como en películas con una gran cantidad de dichas armas, consiguiendo una efectividad mayor al 96.5% (Olmos, Tabik, & Herrera, 2018).

Para poder llevar a cabo el aprendizaje de estas complejas y numerosas redes neuronales es necesario una gran capacidad de procesamiento. Gracias a los últimos avances en las GPUs un usuario normal con un buen ordenador ya puede entrenar un clasificador de Machine Learning en su casa. Sin embargo, la cosa es diferente cuando queremos intentar entrenar algo con Deep Learning.

La tecnología necesaria para conseguir un clasificador decente de Deep Learning es de muy alto nivel y se necesita mucho tiempo y recursos, por lo que a día a hoy es muy complicado realizar uno. A pesar de ello, se tiene esperanzas a que en el futuro sea bastante más accesible debido al rápido avance de los procesadores y de la optimización de los datos.

No obstante, antes de efectuar el procesamiento de imágenes, se debe realizar su captura. Dependiendo de lo que se necesite capturar, se pueden utilizar diferentes tipos de cámara. En este proyecto es necesaria una cámara con sensor de profundidad para obtener la posición tridimensional en el espacio del objeto a identificar, por lo tanto, las cámaras 2D estarían descartadas.

2.3. Cámaras de visión con profundidad

Las cámaras de visión son seguramente el elemento más importante de la visión artificial debido a que sin alguna imagen del entorno no se puede extraer nada de información del exterior para procesarla.

De entre todos los tipos cámaras que existen actualmente, en este proyecto se trabajan con cámaras de visión que también permiten obtener profundidad, llamadas cámaras RGB-D.

Con las cámaras RGB-D se obtiene tanto la imagen a color (en el espacio de color RGB) como la profundidad para cada uno de los píxeles de la imagen obtenida. Este tipo de cámaras son utilizadas en diferentes áreas como en realidad virtual, mapping 3D, reconstrucción o interfaces de usuario.

Los dispositivos y la tecnología de las cámaras de visión con profundidad cambiaron enormemente en el año 2010 cuando Microsoft presentó una nueva cámara con la que jugar a sus juegos, la Kinect, que gracias a un sensor de infrarrojos para obtener profundidad que incluía permitía a los jugadores jugar a los juegos sin necesidad de mandos.

Antes de dicha presentación ya existían otras cámaras de visión como la Swiss Ranger SR4000 o PMD Tech products CamCube 2.0. Sin embargo, eran cámaras que valían demasiado, alrededor de los 10.000$, mientras que la Kinect y las cámaras posteriores que salieron debido a esta llegaron al mercado costando unos 200$, haciendo que sean bastante más accesibles para los usuarios que querían trabajar con dicha tecnología (Litomisky, 2012).

Ilustración 9 – SwissRanger SR400 (izquierda) y PMD Tech products CamCube 2.0 (derecha)

Además de la Kinect de Microsoft existe otra cámara RGB-D que se utiliza bastante para investigación, la Asus Xtion Pro, las cuales obtienen la profundidad del entorno a través del mismo principio.

Estas dos cámaras emiten de uno de sus sensores un haz láser infrarrojo que proyecta una nube o constelación de puntos. En la siguiente imagen, obtenida a través de una cámara infrarroja, se puede apreciar los puntos infrarrojos proyectados por la cámara (Mathe, 2012).

Ilustración 10 – Ejemplo de un haz infrarrojo

Justo después otro sensor de la cámara se encarga de captar los puntos que rebotan en los objetos (siendo su funcionamiento igual que el sónar) para que posteriormente a través del software se calcule la profundidad de cada punto (Mathe, 2012).

A continuación, se analizarán las características de ambas cámaras por separado para comprobar cuál es la que mejor se adecúa al proyecto.

2.3.1. Microsoft Kinect

Desarrollada en noviembre de 2010 como un nuevo periférico de la consola de sobremesa de Microsoft, la Xbox 360. Esta cámara permitía a los jugadores controlar e interactuar con los juegos sin necesidad de mandos, únicamente moviendo su cuerpo.

Ilustración 11 – Kinect

Su uso para los ordenadores data de unos meses más tarde, en junio de 2011, cuando se hizo compatible con Windows y se liberó el SDK (Kit de desarrollo del Software) del dispositivo, permitiendo entonces usar la cámara para investigación.

Entre sus características técnicas destacan:

- Resolución de la imagen RGB: de 640x480
- Resolución de la imagen de profundidad: 320x240
- Rango de alcance: 0.8 - 4m
- Campo de visión: 57.5º de horizontal y 43.5º de vertical

Además, cuenta con un motor incluido dentro de la cámara para poder modificar la posición del ángulo de la cámara de forma remota (iPisoft, 2013). Necesita estar conectada a la red de corriente para funcionar.

Aunque actualmente es una de las cámaras más utilizadas para visión artificial, Microsoft dejó de fabricarla y venderla, tanto la primera versión que es la mencionada en este apartado como la evolución que tuvo, en el año 2017, tras no obtener los beneficios esperados por la compañía.

2.3.2. Asus Xtion Pro-Live

A mediados del año 2012 la compañía china, ASUS, llegó a un acuerdo con Prime Sense, los creadores de los sensores de la tecnología en la que se basa la cámara Kinect, dándole derecho de diseñar y sacar al mercado su propia cámara de visión con profundidad, la Xtion.

Xtion ha tenido varios modelos, pero el más utilizado actualmente es la versión Pro-Live ya que es compatible con más softwares y por lo tanto es un dispositivo más completo. El resto de las especificaciones no cambian respecto a la versión Xtion.

Aunque es compatible con juegos, el uso principal de esta cámara es la investigación en la visión artificial, sin embargo, la cámara de Microsoft cuenta a día de hoy con un mayor número de drivers, lo que la hace más compatible que la Asus.

Ilustración 12 – Asus Xtion Pro-Live

Entre sus características técnicas destacan:

- Resolución de la imagen RGB: de 1280x1024
- Resolución de la imagen de profundidad: 640x480 a 30fps o 320x240 a 60fps
- Rango de alcance: 0.8 – 3.5m
- Campo de visión: 58º de horizontal y 45º de vertical

Además, a diferencia de la Kinect que necesita estar conectada directamente a la corriente, la Asus Xtion Pro-Live se puede utilizar solo con un cable USB 2.0 siendo más polivalente de usar (iPisoft, 2013; Asus, 2013).

A continuación, en una tabla resumen veremos las ventajas de cada una de las dos cámaras más utilizadas en visión artificial.

Dispositivo	Ventajas
Kinect	Tiene un motor para mover el ángulo de visón Mayor comunidad tras ella y más drivers
Xtion Pro-Live	No necesita estar conectada a la red Mejor calidad de imagen Pesa menos y es más pequeña Mayor rango de visión.

Tabla 2 – Tabla resumen de las cámaras de visión

Capítulo 3. Recursos utilizados

Para la elaboración del proyecto se han usado diferentes recursos que serán detallados más adelante. Se han dividido los recursos utilizados en tres apartados: humanos, hardware y software.

3.1. Recursos Humanos

El principal recurso humano de este proyecto, así como de todos los proyectos de Fin de Carrera o similar, es el autor de este. Sin embargo, se ha contado con el apoyo y asesoramiento constante por parte del tutor, Juan Hernández.

Adicionalmente, debido a tratarse de un proyecto del grupo de investigación de la Universidad Carlos III, RoboticsLab, también se ha obtenido ayuda de diferentes personas del laboratorio.

3.2. Hardware

El principal elemento de este proyecto es el robot humanoide TEO (Task Environment Operator) del laboratorio de investigación de la Carlos III, que es donde está situada la cámara y donde más adelante se utilizará mi software para manipular cajas.

TEO fue diseñado para parecerse al cuerpo humano tanto forma (con dos piernas, dos brazos, tronco y cabeza) como en tamaño (1,65m). Actualmente su peso ronda los 65Kgs, pero varía según las modificaciones que se le hagan. Al contrario que otros robots como ASIMO o ATLAS, TEO incorpora las baterías dentro del cuerpo, evitando así tener que llevar una "mochila".

TEO tiene un total de veintiocho grados de libertad, seis en cada pierna y brazos, dos en el cuello para poder mover la cabeza y otros dos en el tronco. Puede levantar hasta un máximo de 2Kgs y puede caminar a una velocidad de 1km/h.

El robot cuenta con 3 procesadores en su cuerpo, uno centrado en las labores de locomoción, otro en las labores de manipulación y otro especializado para la cabeza y la visión.

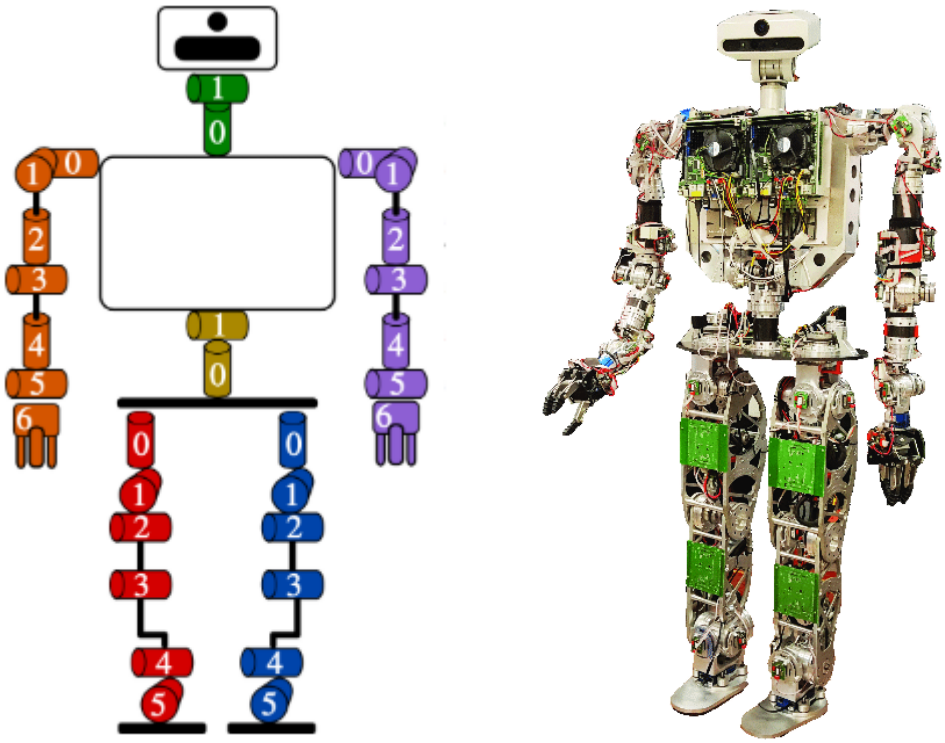

Ilustración 13 – TEO

Sin embargo, para poder optimizar el desarrollo del proyecto se usó la cámara de TEO conectada directamente a un ordenador.

Dicho ordenador fue el portátil de Huawei, Matebook D. Se decidió utilizar un portátil para poder trabajar tanto en casa como en el laboratorio y de esta forma no tener que depender de estar en el laboratorio para desarrollar el algoritmo.

Ilustración 14 – Huawei Matebook D

Sus especificaciones más relevantes son:

- Procesador i5 de séptima generación.
- Tarjeta gráfica dedicada GTX 940M 2Gbs.
- 8Gbs de RAM.

Al ordenador se conectó a través de su USB la cámara de visión con sensor de profundidad que tiene TEO situada en la cabeza, la Asus Xtion Pro-Live.

En el capítulo 2, el del *Estado del Arte*, ya se comentaron todos los detalles de la cámara donde se destacaron las siguientes especificaciones:

- Resolución de 1280x1024.
- Campo de visión: 58º de horizontal y 45º de vertical.
- No necesita fuente de alimentación externa.

3.3. Software

El sistema operativo con el que se ha trabajado es la versión 16.04 LTS de Linux Ubuntu. Es un sistema operativo de código abierto orientado a los usuarios con un fuerte enfoque en mejorar la experiencia de usuario.

Ilustración 15 – Logotipo del Sistema Operativo Ubuntu 16.04 LTS

El lenguaje de programación con el que se ha realizado el algoritmo del proyecto ha sido C++ permitiendo una programación orientada a objetos que otros lenguajes como C, en el que está basado, no permiten.

Ilustración 16 – Logotipo del lenguaje de programación C++

El entorno de programación donde se ha compilado el algoritmo del proyecto ha sido QT Creator 5.

Ilustración 17 – Logotipo de QT Creator

Se ha utilizado este entorno de desarrollo debido a la facilidad de compatibilidad con los proyectos basados en CMake utilizados en los paquetes de OpenCV (que será explicado en este mismo apartado) y además por haberlo utilizado en alguna asignatura de la carrera por lo que era un programa conocido.

Sin embargo, este proyecto no se hubiese podido realizar sin una librería de C++ con la que se pueden utilizar todas las funciones necesarias de visión artificial, OpenCV.

3.3.1 OpenCV

La librería de código abierto OpenCV (OpenCV, 2018) proporciona un entorno de trabajo para el desarrollo de aplicaciones de visión por ordenador en tiempo real. Se puede utilizar tanto con el lenguaje de programación C como con C++ y Phyton. Es una librería multiplataforma capaz de compilarse en diferentes sistemas operativos como Linux, Windows y Mac OS X. (Bradski & Kaehler, 2008; Arévalo, 2004)

Ilustración 18 – Logo de OpenCV

OpenCV facilita enormemente el proceso de aprendizaje e implementación de distintos métodos de visión por computación aislando al programador de las dificultades de estos. De esta forma, esta librería pone a disposición una infraestructura de visión simple de usar que ayuda al usuario a crear sofisticadas aplicaciones de visión de manera rápida (Arévalo, 2004; Bradski & Kaehler, 2008).

La librería de OpenCV contiene más de 500 diferentes funciones que abarca bastantes áreas dentro de la visión, entre las que se incluyen las siguientes:

- Inspección de productos en una fábrica.
- Procesamiento para imágenes médicas.
- Seguridad.
- Robótica.
- Visión estéreo.
- Tratamiento de imágenes con profundidad.
- Machine Learning

Sin embargo, OpenCV tiene unos módulos extra llamadas OpenCV Contrib que incluye funciones más avanzadas que necesitan mayor procesamiento y que no están en las versiones estables totalmente probadas (OpenCV, 2018).

Dichas funciones pueden ser como reconocimiento de patrones, identificación de textos o nubes de puntos para desarrollo de formas en tres dimensiones.

Aunque para el proyecto inicial no fue necesario, si se realizaron funciones adicionales que necesitaron de estos módulos extra. Adicionalmente, se utilizaron unas librerías compatibles pero externas a OpenCV para mejorar la solución final. Dichas librerías son Tesseract (Tesseract-OCR, 2018) y Leptonica (Bloomberg, 2018).

Ambas librerías son utilizadas para complementar la detección de texto incluida en los módulos extras de OpenCV. Con ella, además de una detección bastante precisa, también se consigue una identificación en numerosos idiomas, pudiendo elegir el usuario en qué idioma quiere la identificación.

Ilustración 19 – Logos de Tesseract-OCR y Leptonica

3.4. Presupuesto

En este apartado se detallará el presupuesto utilizado para la elaboración del proyecto teniendo en cuenta los 3 tipos de recursos utilizados.

Referente al hardware, debido a que el precio de TEO no se conoce con exactitud al estar en continua evolución, solo se tendrá en cuenta en el presupuesto el precio de la cámara al haberse utilizado únicamente esta.

Todo el software que se ha utilizado en el proyecto es opensource (de código abierto y gratis de usar para todos los usuarios) por lo que el presupuesto en este apartado será cero.

Finalmente, para el coste humano se realizará el presupuesto acorde al sueldo que recibe aproximadamente un recién graduado de Ingeniería, utilizando 15€/h para el sueldo de un ingeniero junior.

Las horas utilizadas para realizar el proyecto, al no haber llevado a cabo un estudio de ellas se elegirán según el plan europeo. Cada crédito ECST (European Credit Transfer System) constituye a 25 horas de trabajo del estudiante, por lo que, teniendo en cuenta que el Trabajo Fin de Grado son un total de 12 créditos se habrían utilizado un total de 300 horas.

En la siguiente tabla se puede apreciar el desglose del presupuesto de los componentes utilizados para realización del proyecto.

DESCRIPCIÓN	MEDICIÓN	P.U. (€)	P.T. (€)
CAPÍTULO 1 - HARDWARE			
Portatil Huawei Matebook D Portatil utilizado para la realización del proyecto.	1	599€	599€
Asus Xtion Pro-Live Camara de visión con sensores infrarrojos para obtener también la profundidad	1	224,71€	224,71€
CAPÍTULO 2 - SOFTWARE			
Ubuntu Linux Sistema Operativo	0	0€	0€
Qt Creator Entorno de programación	0	0€	0€
OpenCV Librerías de visión	0	0€	0€
OpenCV contrib Módulos extras de visión	0	0€	0€
Tesseract-OCR Librería externa para identificación de caracteres	0	0€	0€
Leptonica Librería externa para identificación de caracteres	0	0€	0€
CAPÍTULO 3 - RECURSOS HUMANOS			
Mano de Obra Carlos Justo de Frías. Graduado en el Grado de Ingeniería Electrónica Industrial y Automática.	300	15€	4500€

Tabla 3 – Presupuesto del proyecto

RESUMEN PRESUPUESTO:

Capítulo 1: Hardware	823,71 €
Capítulo 2: Software	0 €
Capítulo 3: Recursos Humanos	4500,0 €
TOTAL	**5.373,71€**

El presupuesto total para la realización de este Proyecto asciende a:

"Siete mil quinientos setenta y tres euros con setenta y un céntimos"

Capítulo 4. Diseño e implementación del problema

El objetivo principal del proyecto es la correcta identificación en el espacio de una caja para su posterior manipulación por el robot humanoide del laboratorio, TEO. Para ello, se ha utilizado la cámara de profundidad que dispone en su cabeza. TEO es una plataforma de investigación que es utilizada para distintas líneas de proyectos. Por ese motivo, se instaló en un portátil las librerías y recursos base para trabajar en el campo de la visión y se ha ido probando ahí los distintos puntos del proyecto.

4.1. Desarrollo del proyecto

Al principio del desarrollo se probaron diferentes métodos de identificación de objetos que tras diferentes pruebas se acabaron descartando. Tras estas iteraciones, se acabó utilizando, en una primera fase, diferentes operaciones secuenciales sobre imágenes de cajas (en dos dimensiones) realizadas personalmente y encontradas en bases de datos online.

Una vez obtenida toda la información posible de las imágenes en dos dimensiones se siguió el proyecto en el laboratorio con la cámara de profundidad de TEO. Primero se optimizaron los métodos utilizados en dos dimensiones para que funcionara bien con el cambio de cámara y después se realizó una calibración de la cámara, necesaria para poder situar correctamente la caja en el espacio.

Tras la calibración, se calculó la distancia real a la que se encuentran los vértices de la caja utilizando la información obtenida y procesada previamente. Con las dimensiones exteriores reales de la caja ya conocidas, se obtienen los puntos que TEO utilizará para poder manipularla.

A continuación, se explicará detalladamente y por separado como se ha llevado a cabo cada una de las fases del proyecto.

4.1.1 Procesamiento con imágenes en 2D

Para la identificación de objetos se pensó utilizar, en la primera fase, el proceso que más se ha usado en los últimos años para detectar objetos, *Machine Learning*. Es cierto que hay otro proceso que funciona mejor: el *Deep Learning,* pero conlleva un coste computacional mucho mayor y una base de datos bastante grande debido a que el mismo programa decidiría qué es caja y qué no de todas las muestras. Sin embargo, en el *Machine Learning* el programador decide qué es caja y el detector aprende, por lo que no se intentó la opción anterior (Restrepo Arteaga, 2015).

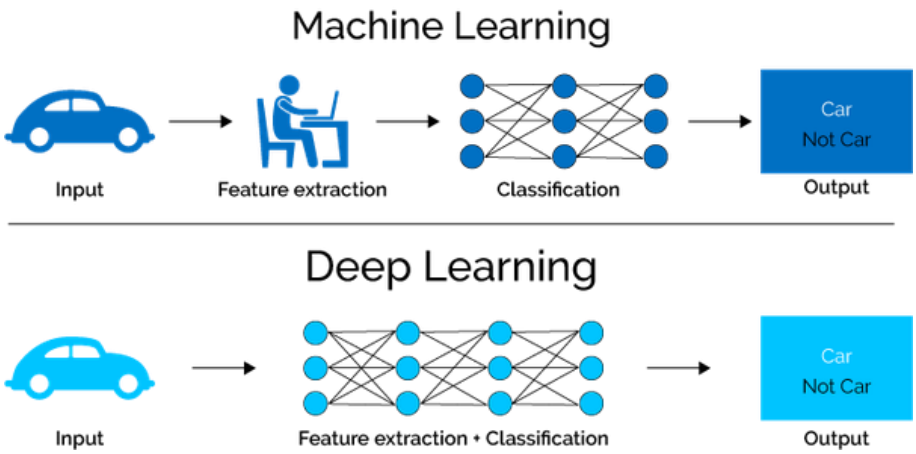

Ilustración 20 – Diferencias entre Machine Learning y Deep Learning

Entre todos los tipos de *Machine Learning* que existen, se probó primero, un *Machine Learning* basado en Bayes, en el que había que "enseñar" a un detector introduciendo descriptores (datos característicos, normalmente numéricos) del objeto a detectar. El problema surgió al intentar elegir estos descriptores.

En un primer lugar se intentó incluir los descriptores de perímetro y área, pero existía el inconveniente de que estos valores varían mucho según el tamaño y la orientación de la caja, así como a la distancia a la que estuviera de la cámara. También se pensó en el usar el grafo de la caja como descriptor, pero no se acabó utilizando debido al gran coste computacional que presenta, sumado al coste del Machine Learning con Bayes. Los únicos descriptores que se podrían utilizar eran el color, la rectangularidad, y la circularidad. Sin embargo, al intentar procesar estos descriptores, los datos variaban mucho dependiendo de cómo estuviera orientada la caja sobre la cámara.

Por todo esto se acabó descartando un *Machine Learning* basado en Bayes, y se probó posteriormente un *Machine Learning* basado en los gradientes de las imágenes, el *Histogram of Oriented Gradients* (HOG).

En este nuevo detector, en lugar de introducir las características de las cajas, se deben introducir imágenes positivas y negativas. Las imágenes positivas se introducen en primer lugar para "indicarle" al detector que esas cajas son las cajas que va a tener que detectar, o similares. A continuación, se introducen imágenes negativas, que suelen ser objetos que podrían aparecer con las cajas, para "indicarle" en este caso que estos objetos no son cajas, y que no debe detectarlos como tal.

Las imágenes positivas debían ser imágenes cuadradas obligatoriamente, del mismo tamaño y donde únicamente apareciera la caja. Para realizar estas pruebas se utilizaron los recortes de 3 cajas, con un total de más de 300 imágenes de cada, obteniendo un total de aproximadamente 1000 imágenes positivas.

Ilustración 21 – Los tres tipos de cajas utilizadas para HOG

Al principio, se intentó entrenar el detector con unas imágenes positivas de 240x240px y unas imágenes negativas de 1920x1080px, pero daba problemas de memoria y saltaba una interrupción en el programa, por lo que se tuvo que reducir la resolución de todas las imágenes para disminuir la memoria total.

El tamaño definitivo de las imágenes fue de 120x120px para las positivas y de 848x480px para las negativas.

Con estos últimos valores se consiguió, finalmente, entrenar un detector. Sin embargo, la detección no fue tan buena como se esperaba. Las sombras de las cajas formaban parte de la detección, incluso a veces daba como positivo una sombra aislada sin caja, como se puede apreciar en las siguientes imágenes.

Ilustración 22 – Errores en la detección

Para intentar solucionar este inconveniente, se rehicieron todas las fotografías de las cajas, pero con una iluminación lo más homogénea posible. Se utilizaron distintos *softbox* para conseguir que se proyectaran las mínimas sombras posibles.

Después, se tuvieron que realizar de nuevo los recortes positivos (como aparece en la siguiente imagen), manteniendo los mismos recortes negativos.

Ilustración 23 – Nuevas fotografías de las cajas usando Softboxes

Una vez recortadas todas las nuevas imágenes, se volvió a entrenar el detector, ya obteniendo una salida donde solo se detectaba la caja y no la sombra, pero aun así detectaba falsos positivos frecuentemente, por lo que el resultado obtenido no era lo esperado y no se podía utilizar. Se acabó desechando este método.

Ilustración 24 – Errores en la detección final

Se comenzó a probar otros métodos y, para intentar evitar los errores de fondo, se empezó a experimentar con imágenes en 2D de cajas, con el fondo liso y cambiando un poco el orden de obtener los datos. En primer lugar, se obtendrían los datos necesarios tanto en 2D

como en 3D de la caja, con fondo uniforme, para, una vez obtenidos los datos necesarios, intentar implementar un filtro para conseguir eliminar el fondo.

Estas imágenes sin fondo se obtuvieron del buscador de imágenes de Google y sobre estas se realizaron diferentes operaciones para intentar obtener las características de las cajas.

En primer lugar, se obtuvieron los bordes de las cajas para, a continuación, intentar obtener información de la caja. Existen dos algoritmos principales para obtener los bordes de una imagen: Sobel y Canny. Se experimentó con ambos para comprobar con cual se obtenían mejores resultados y se obtuvieron las siguientes salidas:

Ilustración 25 – Diferencias entre Canny (izquierda) y Sobel (derecha)

Una vez obtenidos los resultados con ambos métodos y, tras analizar ambas imágenes, se llegó a la conclusión de que con el algoritmo de Canny las líneas obtenidas son más fieles a las originales. Es decir, que están colocadas exactamente dónde se sitúan los bordes reales.

Además, la salida de Canny es una imagen binaria donde lo blanco son las líneas y en negro aparece todo lo que no son líneas, mientras que, con el método de Sobel, se obtienen las líneas en una escala de grises con varios píxeles de grosor, difiriendo del de Canny que el borde se representa con solo un píxel.

Después de elegir el método para detectar los bordes de la imagen, se estudió qué característica de la caja se debería obtener primero. Entre las características que mejor podían definir mejor a una caja se eligieron dos: el color y la forma cubica donde todas sus aristas son rectas.

Debido a que no todas las cajas tienen el mismo color, se prefirió comenzar obteniendo las líneas de la imagen porque que todas sus aristas sean rectas es una característica que todas las cajas tienen en común.

Para obtener las rectas de una imagen se utilizó la transformada de Hough (1981), que busca en toda la imagen las figuras geométricas que más se parezcan a una recta, teniendo que ser la imagen una imagen binaria, por lo que el algoritmo de Canny previamente elegido fue la mejor opción. Se puede utilizar la transformada de Hough para detectar círculos, rectas y curvas. En este proyecto se utilizó el método de detectar rectas.

La transformada de Hough para rectas tiene 2 posibles métodos, HoughLines y HougLinesP. Las diferencias teóricas entre los dos es que en el HoughLines las rectas que se obtienen son rectas infinitas mientras que las rectas obtenidas con el método HoughLinesP son rectas finitas, donde se obtienen el punto inicial y final de cada recta (Bradski & Kaehler, 2008; OpenCV, Hough Line Transform, 2018). A continuación se muestran las diferencias en la detección de líneas en la misma imagen.

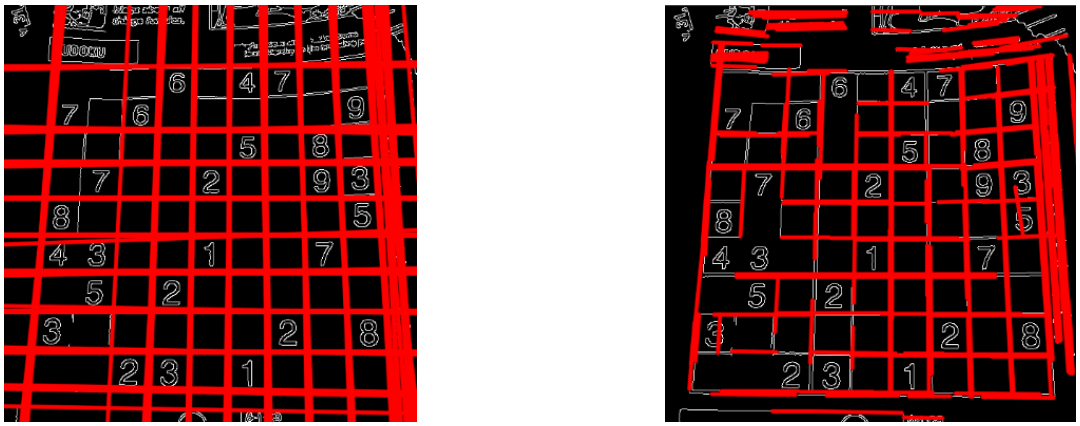

Ilustración 26 – Diferencias teóricas entre HoughLines (izquierda) y HoughLinesP (derecha)

Teóricamente, según las imágenes de los documentos de OpenCV, HoughLines es el método con el que se obtienen mejores resultados, pero, para estar más seguros, se probaron ambos métodos con las imágenes de cajas del proyecto para decidir cuál utilizar, obteniendo los siguientes resultados.

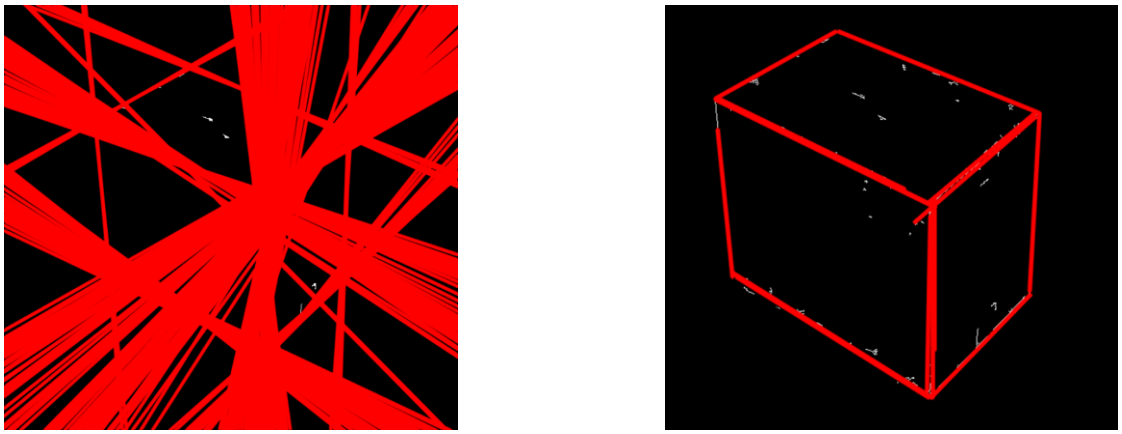

Ilustración 27 – Diferencias reales entre HoughLines (izquierda) y HoughLinesP (derecha)

A pesar de que se esperaba que fuera mejor la opción, la detección de líneas infinitas no acabó siendo la más adecuada. Las líneas de Canny no son siempre totalmente rectas y para una misma línea se detectan bastantes rectas, haciendo que se pierda mucha información como se puede ver en la imagen de la derecha.

Se intentó solucionar la detección modificando los valores tanto del Canny como del HoughLines, pero no se consiguieron resultados lo suficientemente buenos como para considerar este método como posible y se acabó descartando.

Finalmente se utilizó el método HoughLinesP, donde se obtenía unas rectas que no hacía que se perdiera información y que estaban colocadas únicamente encima de los bordes. El único inconveniente de este método es que las líneas que detectaba no tenían la misma longitud que el borde real ya que eran más cortos en la mayoría de los casos, dejando como un vacío en las esquinas la mayoría de los casos (círculos amarillos de la siguiente imagen). Además, en varias aristas de la caja se detectaban varios bordes donde debería haber solo uno (círculos azules).

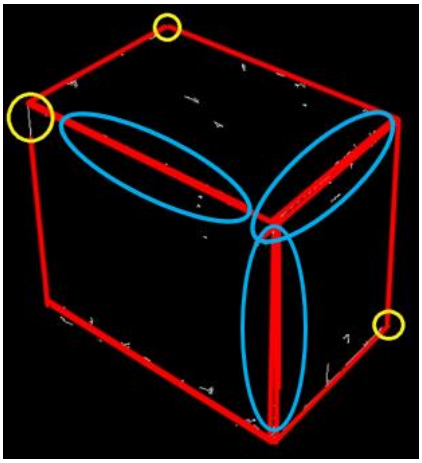

Ilustración 28 – Errores en la detección de líneas en las imágenes

Para intentar solucionar que no se juntasen los bordes se probó una dilatación de la imagen binaria que se obtenía con Canny, para posteriormente, volver a buscar las líneas de la imagen, logrando que todas las líneas estuviesen conectadas.

Pero esta solución aumentaba el problema de detectar muchas líneas para un solo borde, como se aprecia en la siguiente imagen, obteniendo una cantidad de puntos demasiado grande.

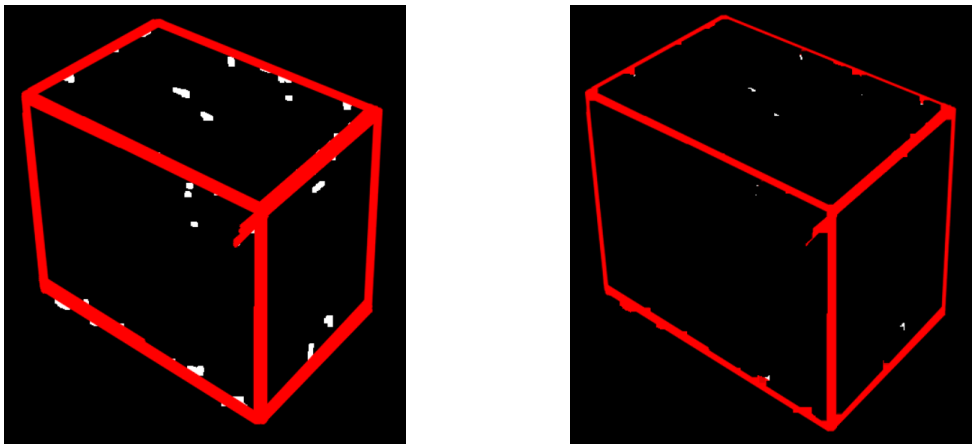

Ilustración 29 – Posibles soluciones a los problemas de detección de líneas

Se intentó solucionar utilizando en la imagen la transformación inversa a la dilatación, la erosión, pero el único cambio que se producía era reducir visualmente las líneas y no disminuía el número de rectas, por lo que no servía. Aunque es cierto que con esta última operación quedaban muy diferenciadas las caras de la caja, por lo que se empezó a intentar obtener cuadrados en dichas caras.

Para obtener dichos cuadrados, se buscaron diferentes métodos que pudiesen realizarlo, pero el que mejores resultados obtenía era un buscador de contornos que buscaba figuras geométricas cuadradas. En un principio solo encontraba cuadrados muy exactos, con ángulos muy parecidos a 90º, pero como la detección era baja, se decidió modificar esta condición, buscando cuadrados con ángulos más alejados de 90º. Los cuadrados obtenidos aportaron unos resultados muy buenos, como se puede apreciar en las siguientes fotografías.

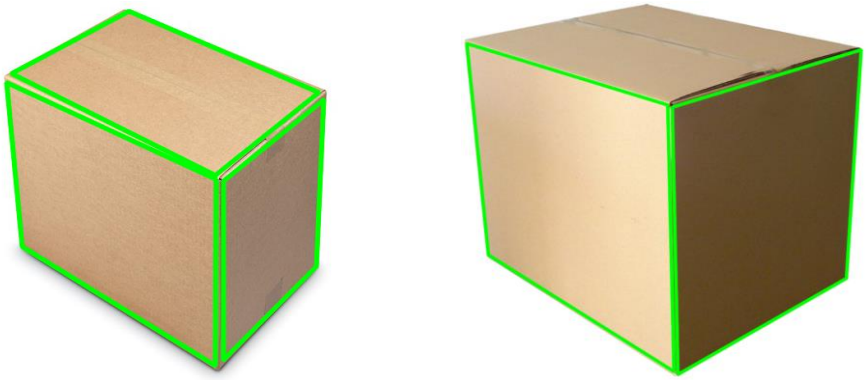

Ilustración 30 – Ejemplos del buscador de cuadrados

Los cuadrados obtenidos coinciden con bastante exactitud con las caras de las cajas. Sin embargo, no siempre se detectaban los tres cuadrados de los lados, debido a que no se obtenían correctamente las líneas de todas las aristas en los pasos anteriores. No obstante, este “error” no es algo que afecte demasiado porque con solo los cuadrados de dos caras se pueden obtener

los tres planos de las cajas y, con ello, todos los puntos que necesarios para obtener sus características.

Por ejemplo, si se han obtenido los tres cuadrados de la caja, es obvio que se tiene los planos necesarios para definirla, pero en el caso de que solo se hubieran detectado dos cuadrados, no habría ningún problema, ya que se tendría tres puntos del cuadrado que no se ha identificado, con los que se podría obtener el plano restante.

Tras los resultados apreciados en la ilustración 30, se decidió identificar los puntos de las esquinas para intentar obtener los siete puntos característicos de los vértices de la caja. De los cuadrados anteriores, se extrajeron los puntos que se usaron para dibujarlos. Al principio se obtuvieron demasiados puntos, ya que había algunos que se repetían varias veces. Con un filtro de posición que comprobaba la colocación de cada punto respecto a los demás se consiguió eliminar los puntos repetidos, quedando en la imagen solo los puntos que ofrecían una posición e información diferente con el resto.

Aun así, dado que cada cara de la caja está representada por un cuadrado, en las esquinas donde coincidían más de un cuadrado había más de un punto significativo. De esta forma en las esquinas donde coincidían dos cuadrados había dos puntos y en la esquina donde coincidían tres esquinas había tres.

Con el fin de conseguir uno único punto por vértice, se calculó la media entre los puntos más cercanos entre sí para obtener un único dato con el que operar, siendo un dato más exacto al estar el nuevo punto más cercano a la esquina real de la caja.

Ilustración 31 – Ejemplos de la obtención de los puntos de las esquinas de una imagen

Una vez obtenidos los puntos medios y antes de seguir el proyecto en el laboratorio, dado que no existía un patrón constante en la detección de los puntos de una imagen a otra (De

un *frame* a otro cambiaría el orden), se ordenaron los puntos de tal forma que siempre tuvieran la misma colocación.

Con una ordenación de los puntos se simplificaría de gran manera los procesos posteriores, siendo universales para todos los casos, y no teniendo que comprobar a cada rato como están colocados los puntos para operar con ellos.

Para el primer punto de la ordenación se eligió el punto central de la caja, aquel que pertenece a los tres planos. El siguiente punto de la ordenación sería el colocado más alto entre los dos puntos situados más a la derecha, para después colocar el otro de los dos puntos, el que está más abajo. Una vez añadido estos tres, se seguiría una ordenación en sentido horario, es decir, el siguiente es el que está colocado más abajo del todo, después, los dos más a la izquierda – primero el de abajo y luego el de arriba - y finalmente el que está más arriba entre los siete puntos. Toda la ordenación de puntos se puede apreciar en la siguiente imagen.

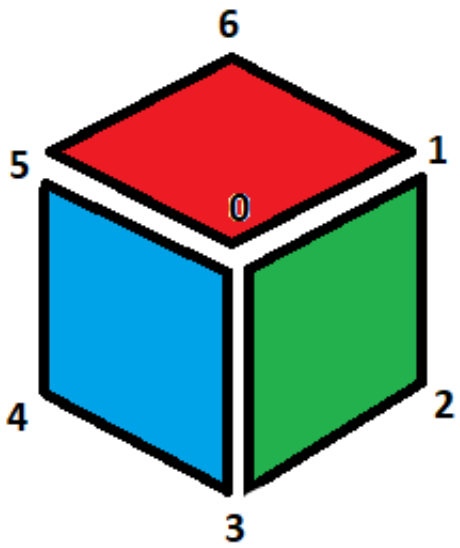

Ilustración 32 – Ordenación de los puntos

El caso anterior sería cuando se ha obtenido los cuadrados en las tres caras de la caja, pero si no se tuvieran los 3 lados y solo se tuvieran dos, la ordenación seguiría el mismo orden, primero el punto central, y después en sentido horario empezando por la derecha. Únicamente habría que hacer unas comprobaciones para saber cuál es el lado en el que no se ha obtenido el cuadrado.

Algunos ejemplos de la ordenación de las cajas se pueden ver a continuación.

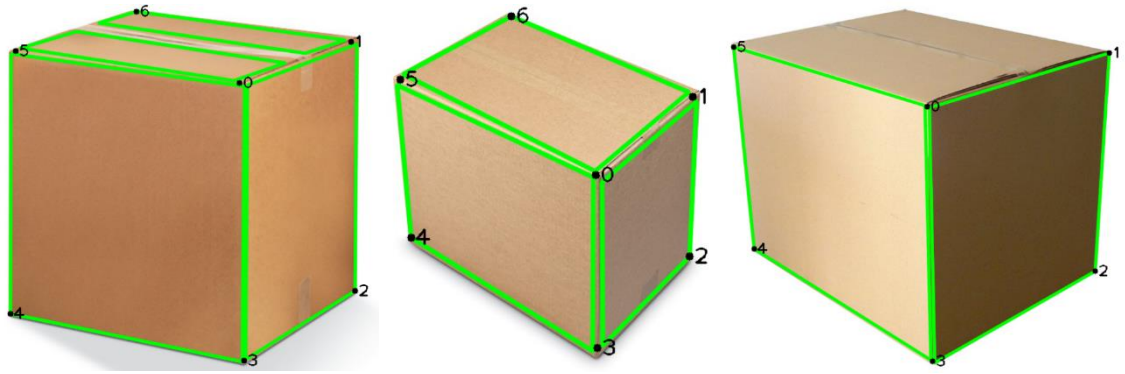

Ilustración 33 – Ejemplos de la ordenación final de los puntos

Los resultados anteriores son bastante buenos, tantos los cuadrados como los puntos medios en las esquinas están bastante bien colocados. Pero siguen siendo imágenes sueltas con buena resolución y con un fondo blanco uniforme. Por ello, antes de intentar trabajar en el laboratorio con imágenes a tiempo real, se intentó primero una segmentación de color para intentar extraer la caja del fondo.

Aunque en un principio se iban a usar varios tipos de caja, debido a querer intentar una segmentación por color para que no aparezca el fondo de la imagen como ruido, se limitaron las cajas únicamente a cajas marrones para simplificar el proceso y para poder hacer esta segmentación de color.

Se intentaron varios espacios de colores, para intentar extraer el color marrón de las cajas, como el RGB, HSV, YCbCr.

El Espacio RGB es el más conocido y utilizado en todo el mundo, en él, cada color es una mezcla de los tres colores primarios, el rojo, el azul y el verde, por lo que es más intuitivo para los seres humanos.

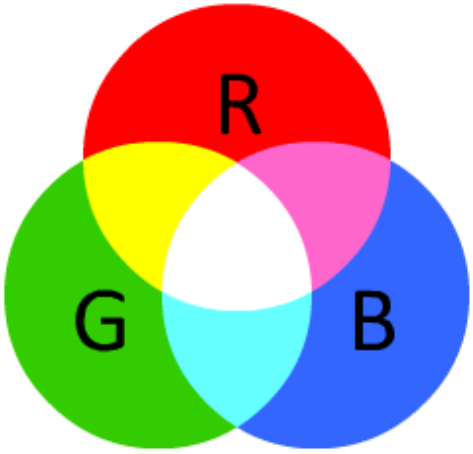

Ilustración 34 – Espacio de color RGB

A diferencia del modelo RGB, el espacio de color HSV sigue una representación más parecida a coordenadas cónicas, siendo una interpretación más parecida a la forma en que los humanos perciben los colores, pues se agrupan las tonalidades de color.

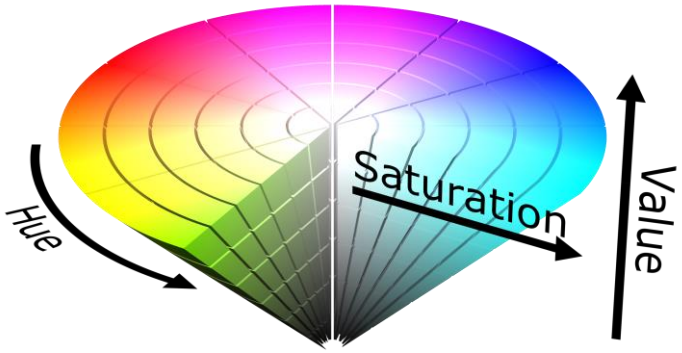

Ilustración 35 – Espacio de color HSV

Por último, el espacio de color YCbCr se basa en la obtención de la luminancia o luminosidad y de la crominancia, el color. No es un espacio de color absoluto, sino una forma de codificar la información del espacio RGB.

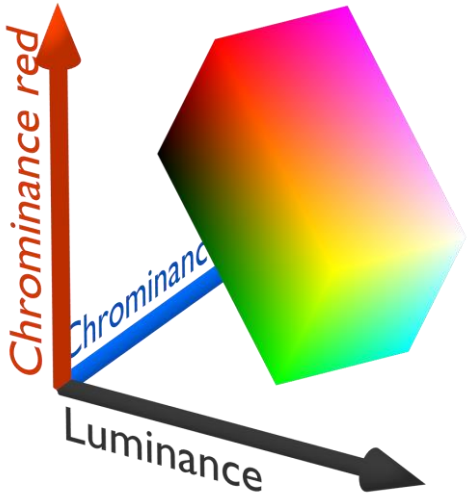

Ilustración 36 – Espacio de color YCbCr

Se empezó probando con el RGB, pero los resultados variaban mucho respecto a los cambios de tonalidad de una cara a otra, por lo que se intentaron los otros dos espacios.

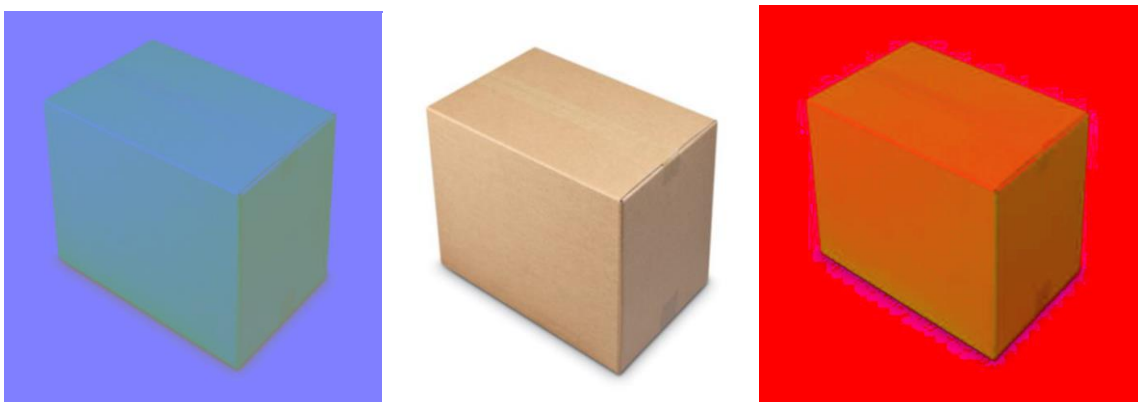

Ilustración 37 – Diferencias entre HSV (izquierda), RGB (medio) y YCbCr (derecha)

Para la separación del color marrón se utilizó un método que nos permitía mostrar en blanco los espacios de la imagen donde aparecía los colores buscados, que, en este caso, eran tonalidades de marrón. Para obtener estos rangos se utilizó en un primer lugar los tonos de la piel, pero se modificaron un poco para adecuarlos mejor al color marrón de las cajas.

Con los rangos ya elegidos, se realizó la operación matemática *AND* sobre las dos imágenes, la real y la binarizada con los sitios donde está situado el color marrón en ambos espacios de color, para mostrar en una imagen nueva únicamente el color marrón detectado.

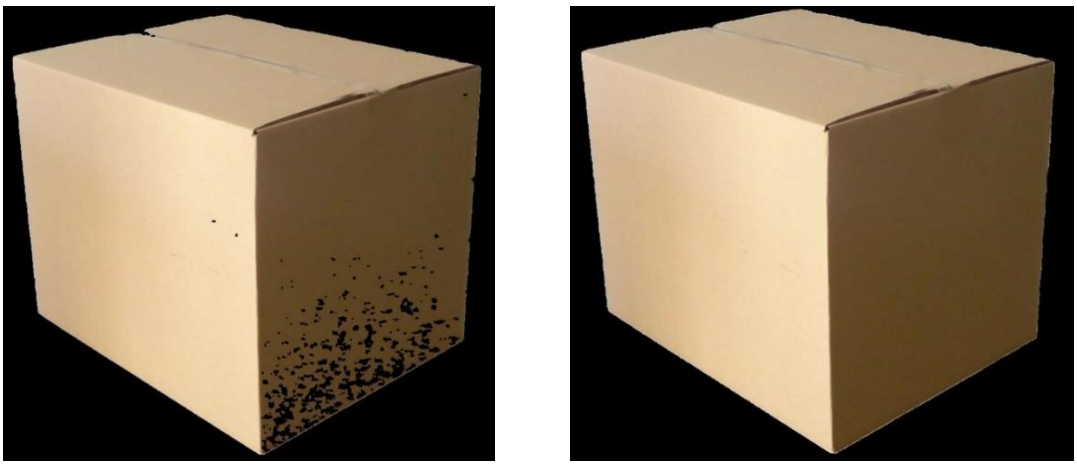

Ilustración 38 – Diferencias en las máscaras utilizando HSV (izquierda) y YCbCr (derecha)

Con ambos se obtenían mejores resultados que con el espacio de color RGB, pero se acabó eligiendo el YCbCr por encima del HSV debido a que en este último el color marrón estaba situado al principio y al final del rango a elegir, por lo que al intentar meter ambas secciones se producirían errores de segmentación.

Aunque parecía que se obtenían buenos resultados, al intentar probar el algoritmo con un video a través de una webcam el resultado fue totalmente opuesto. Según la iluminación que había en la habitación los resultados cambiaban muchísimo (por ejemplo, en el laboratorio con

el cambio de luz de medio día al atardecer, los resultados variaban completamente), llegando a no detectar nada de la caja como marrón. Por lo que la opción de un filtro de color se acabó descartando y se volvieron a incluir cajas de todo tipo de colores dentro de las posibles para ser procesadas.

4.1.2. Procesamiento con imágenes en 2D obtenidas de la cámara de TEO

Una vez desarrollado todo lo anterior, se decidió seguir el desarrollo en el laboratorio con la cámara de TEO para así utilizar imágenes que realmente se iban a utilizar en el resto del proyecto.

Con respecto al filtro del fondo, se decidió colocar un fondo uniforme en el laboratorio para poder avanzar con el proyecto.

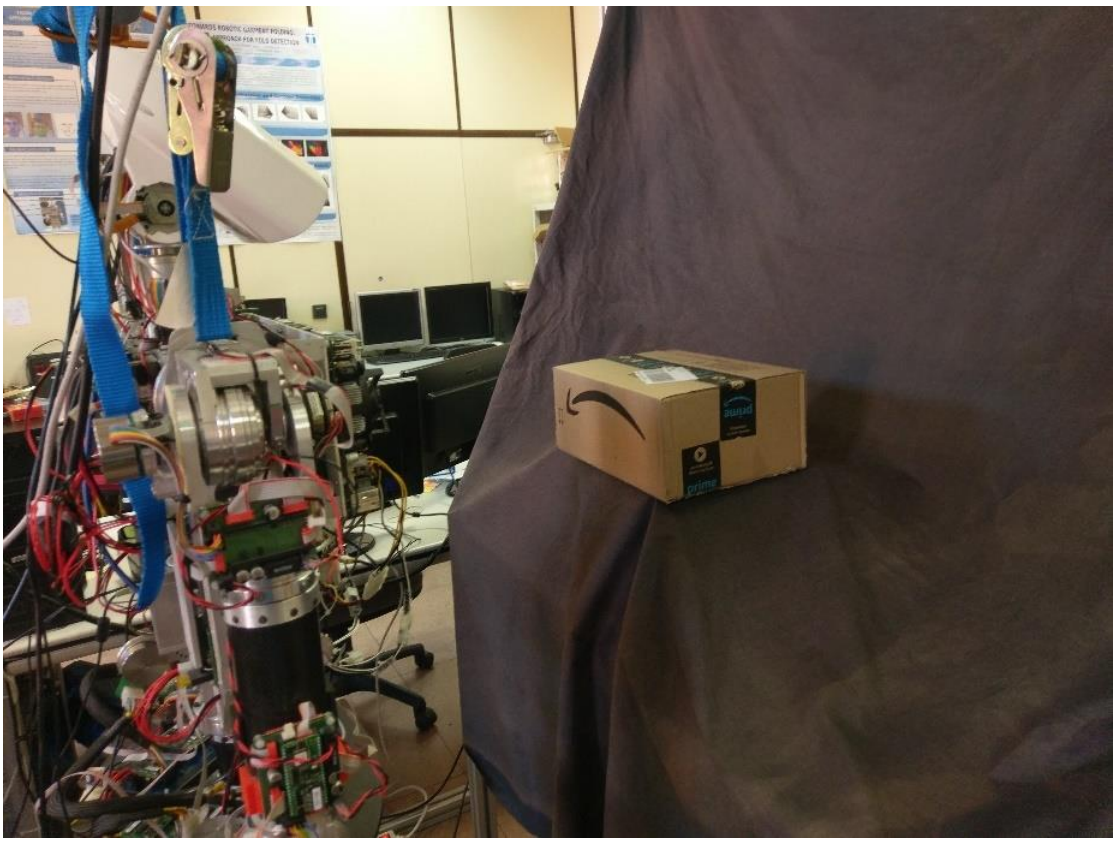

Ilustración 39 – Fondo uniforme del laboratorio

Una vez en el laboratorio se intentó utilizar el algoritmo obtenido anteriormente, sin modificarlo para comprobar como cambiaban los resultados de trabajar con imágenes a trabajar con video a tiempo real, obteniendo unos resultados bastante peores de lo esperado.

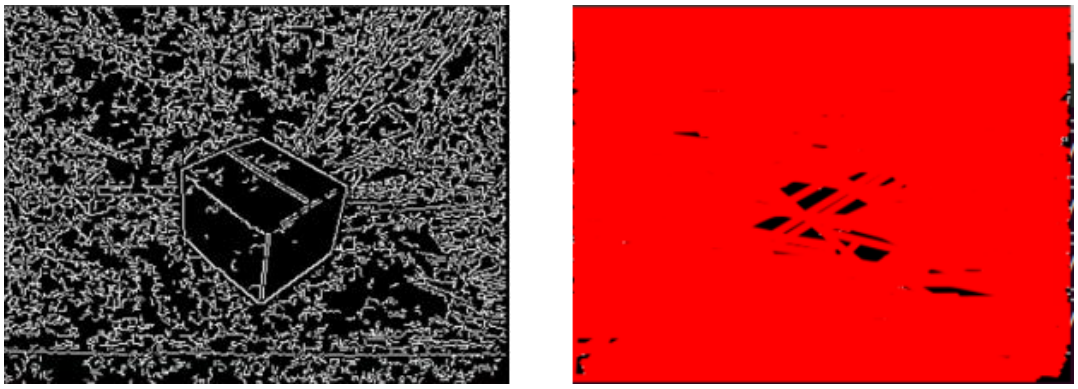

Ilustración 40 – Resultados del procesamiento 2D en el laboratorio

Como se puede apreciar en la imagen, la salida obtenida tras el algoritmo de Canny identifica casi toda la imagen como borde, por lo que posteriormente cuando se detectan las líneas rectas se obtienen bastantes, y al intentar mostrarlas hacía que se perdiera mucha información de la imagen.

Muchos de los bordes que se detectaban son los pliegues de la lona negra utilizada como filtro para eliminar el fondo, así como las diferentes iluminaciones que existen en esos pliegues.

También muchos bordes detectados se deben al ruido gaussiano que se produce por la cámara, que son pequeñas variaciones en el valor numérico de los pixeles, modificándolos de los que realmente debería tener.

Quizás un filtro para intentar eliminar ese ruido haría que se detectaran menos bordes con el método Canny, pero se tendría que usar un filtro muy grande lo que provocaría que se difuminaría demasiado la imagen, y, por tanto, que se perdiera demasiada información que después podría ser útil.

La solución para este inconveniente fue utilizar una segmentación de color, como el intentando anteriormente, pero no para extraer el color marrón, si no para eliminar todo lo que sea parecido al color negro de la imagen.

Debido a que el color que se iba a buscar era el negro, con el espacio de color RGB se podría realizar sin problemas.

En lugar de intentar utilizar los tres canales del espacio RGB, se estudió cada canal por separado para ver en cual habría más diferencia con el marrón, ya que será el color predominante en las cajas, comprobando en cual había mayor diferencia de tonalidad, obteniendo los siguientes resultados.

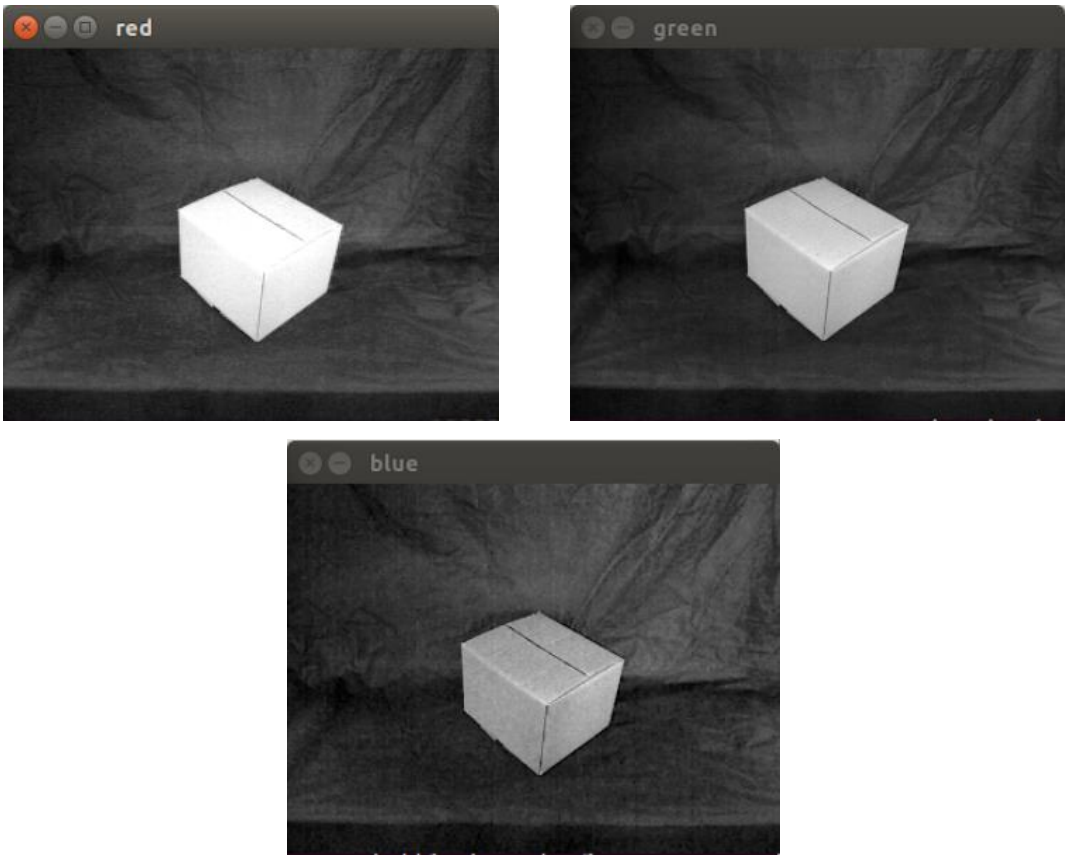

Ilustración 41 – Diferencia entre los tres canales del espacio RGB

Vistos los resultados anteriores, el canal en el que se aprecia mayor contraste entre el fondo y la caja es el rojo, lo cual tiene sentido ya que en el marrón el color mayoritario de los tres es este.

Elegido ya el canal rojo entre los tres, se realizó después una umbralización para obtener una imagen binaria donde solo se aprecié todo aquello que no es negro, obteniendo en blanco el perímetro de la caja, en este caso.

Para evitar un posible ruido en la imagen, se realizaron diferentes operaciones sobre la imagen, utilizando en este caso una transformación de apertura, u *opening*, que es la realización en primer lugar de una erosión en la imagen seguida de una dilatación de igual magnitud.

En las siguientes imágenes se puede ver en imagen de la izquierda una umbralización con ruido y a la derecha una corrección en la imagen tras el uso de la transformación.

Ilustración 42 – Corrección de la máscara

Una vez obtenida ya la máscara de la caja, se aplicó sobre la imagen original la operación matemática *AND*, para así eliminar todo el ruido procedente del fondo, para de esta forma utilizar el algoritmo desarrollado anteriormente.

Sin embargo, una vez juntadas las imágenes, al intentar realizar la detección de bordes y el buscador de cuadrados el resultado no fue lo esperado, ya que no detectaba los cuadrados ni los bordes de la caja bien, principalmente debido al ruido gaussiano que existía en la obtención de la imagen por la cámara.

Debido a esto último, se dejó de lado intentar de identificar los cuadrados a través de la imagen original, y, en su lugar, se empezó a probar a obtener el perímetro de fuera de la caja con la imagen de arriba.

Para obtener el perímetro se volvió a utilizar el detector de líneas HoughLinesP, donde se obtenían en rojo las 6 líneas externas que delimitan el perímetro de la caja.

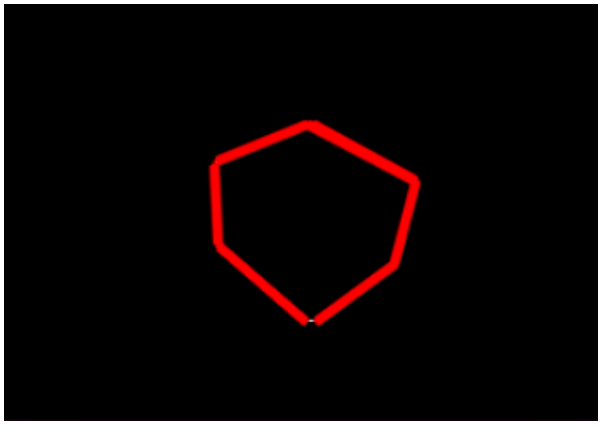

Ilustración 43 – Ejemplo del perímetro de la caja

Además de obtener las rectas con el método HoughLinesP, también se obtenían los dos puntos que definen cada una de las líneas, los puntos de cada extremo, lo que se obtiene 2 puntos en cada esquina de la caja.

Con esos puntos, como ya se hizo anteriormente, se realizó la media entre los puntos que estén más cercanos para tener un único punto en cada esquina.

Aunque, como ya ocurría con las imágenes, los puntos obtenidos están colocados de forma totalmente aleatorias sin existir un patrón definido, como se puede apreciar en la siguiente imagen, por lo que el siguiente paso fue volver a ordenar los puntos.

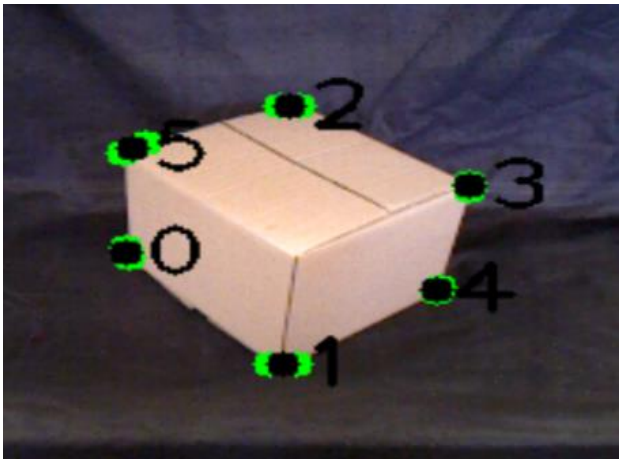

Ilustración 44 – Ejemplo de la obtención de los puntos en el laboratorio

Al contrario que con las imágenes que se obtuvieron en Google donde todas las cajas estaban colocadas de la misma forma, ahora en el laboratorio podría haber dos formas: una primera donde la caja esté un poco girada, viendo tres caras de la caja; o donde una de las caras de la caja estuviera de frente a la cámara y donde solo se verían 2 de las caras.

Los dos tipos de colocaciones se pueden apreciar en las siguientes imágenes.

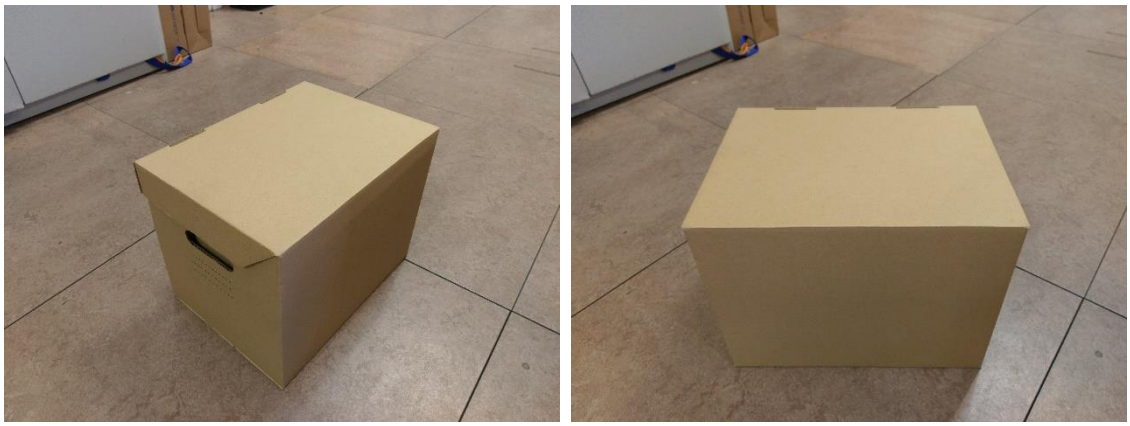

Ilustración 45 – Los tipos de colocación de la caja, girada (izquierda) y de frente (derecha)

Para saber si la caja estaba colocada de frente o de perfil se realizó una comprobación de la posición de los dos puntos situados más abajo. Si la distancia en el eje Y de los dos puntos es muy grande, significa que se está en el primer modo, con la caja girada. Mientras que si la distancia es muy pequeña se encuentra en el segundo modo donde la caja está de frente.

Estas dos formas de colocación de las cajas se tenían que tener en cuenta durante todo el proceso debido a que la ordenación de los puntos y la obtención de las características de la caja se tendrían que calcular de forma diferente si está de una forma u otra. Por esta razón se realiza antes que la ordenación de los puntos.

En la ordenación de puntos del primer caso, el primer número sería el colocado más arriba de la imagen y después se seguiría una ordenación en sentido horario, siendo los siguientes puntos los de la derecha, estando primero el de arriba, después el de abajo y por último los dos de la izquierda.

Para el segundo caso, igual que con el primero, se seguiría una ordenación en sentido horario en la ordenación de los puntos, pero empezando, en este caso, por el punto colocado más a la izquierda de todos. Después irían los dos de arriba, el de la derecha y para terminar los dos de abajo.

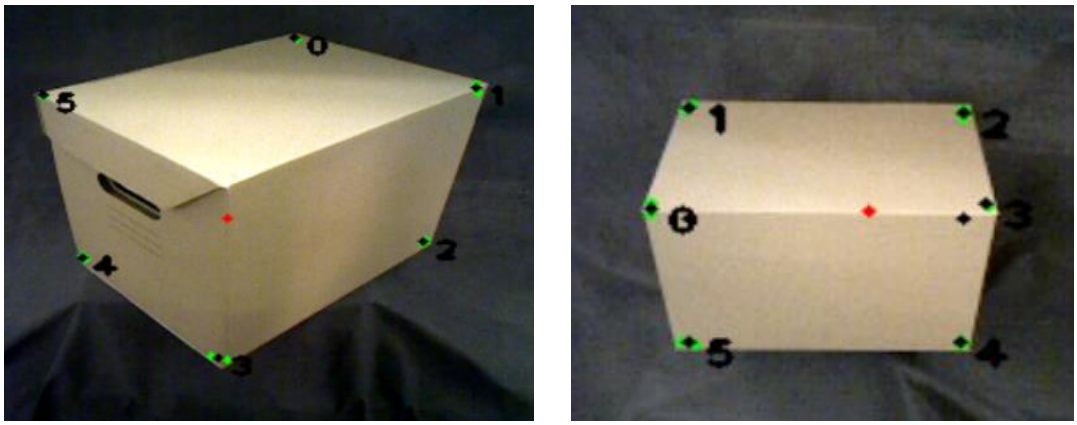

Ilustración 46 – Puntos de las esquinas ordenadas para las dos formas de colocación

Una vez ordenados correctamente todos los puntos en ambos casos, lo siguiente que se realizó fue obtener las distancias de ellos. Las distancias que se obtienen están relacionas con Teo, exactamente respecto al eje de coordenadas situado en la cabeza de TEO, donde se encuentra la cámara de profundidad de Asus.

El eje Z es el eje de profundidad, paralelo al suelo, siendo Z mayor cuando más lejos se esté de la parte frontal de TEO y negativo si estuviera por detrás de TEO. Se puede apreciar en la siguiente imagen como la línea azul.

El eje Y, indicado con la línea verde de la imagen, es el eje de altura.

Finalmente, el eje X, dibujado con la línea roja, indica la posición horizontal del objeto sobre la cámara de TEO.

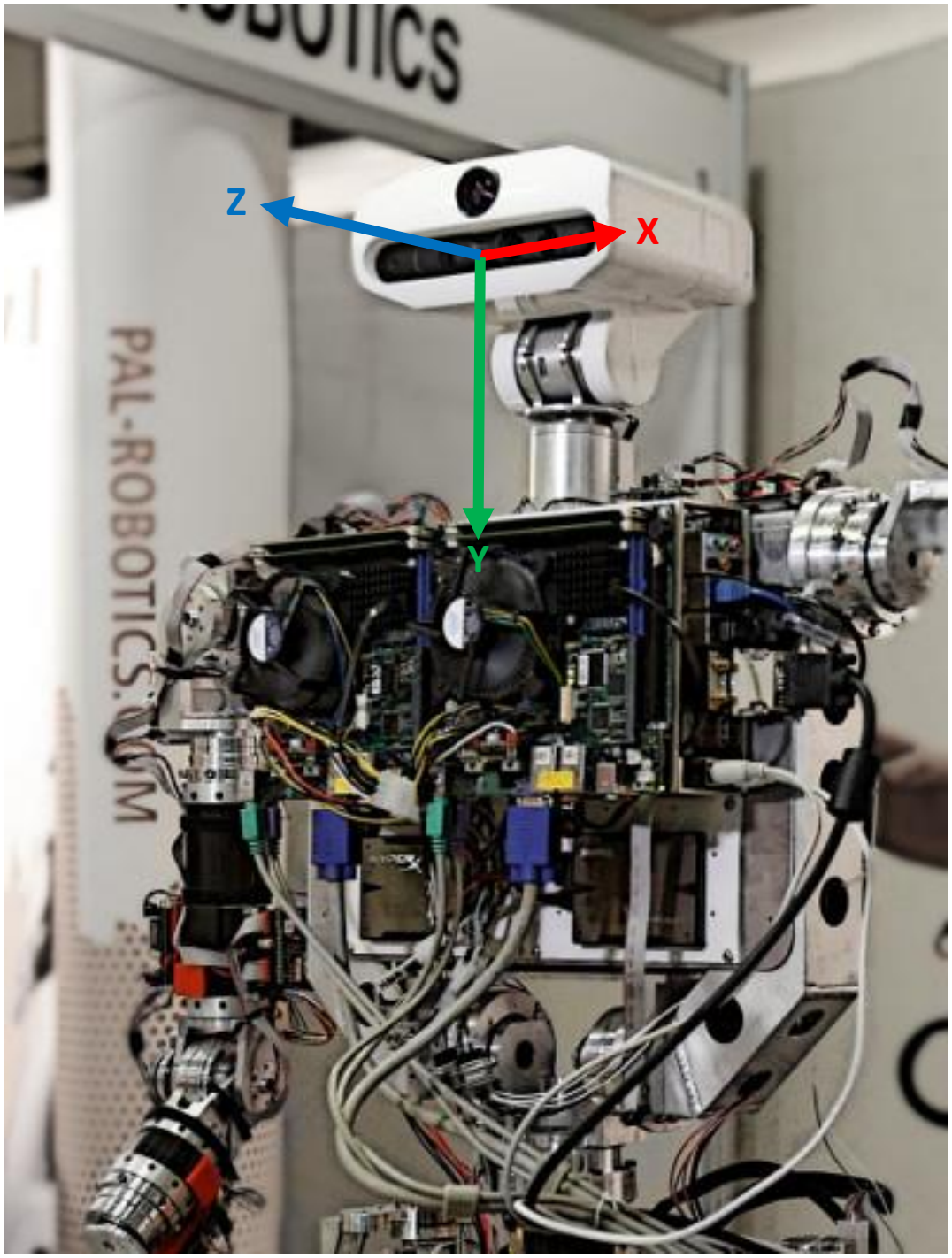

Ilustración 47 – Ejes de coordenadas de TEO

La obtención de los ejes difiere entre unos y otros. El eje X se obtiene a través de una calibración de la cámara obteniendo de este proceso varias gráficas que relacionan los píxeles con la profundidad. Por otra parte, los ejes Y y Z se obtienen a través de la profundidad

realizando sobre estos datos algunas operaciones matemáticas además de una calibración de la cámara similar a la anterior. Debido que se usan técnicas distintas para las coordenadas de la caja, la siguiente parte se divide en dos, una sobre la calibración de la cámara y otra sobre el procesamiento de las imágenes 3D.

La calibración de la cámara permite obtener dos gráficas que relacionan la profundidad de los píxeles. A través de estas gráficas, necesarias para la obtención de los ejes X, Y y Z, se consigue identificar la distancia real en algunos pasos del apartado de procesamiento 3D.

El procesamiento 3D corresponde a la explicación de los métodos utilizados para la obtención de las posiciones X, Y y Z de cada uno de los puntos de la caja, así como de las características geométricas de la caja y los puntos de aproximación que utilizaría TEO para cogerlas.

En los siguientes apartados se definen en profundidad los métodos de calibración de la cámara y procesamiento 3D.

4.1.3. Calibración de la cámara

La calibración de la cámara, como se ha indicado anteriormente, es esencial para la identificación de la distancia real de un punto respecto al centro de la imagen obtenida.

En dicho proceso se obtuvieron dos gráficas relacionadas con la profundidad: una con el eje horizontal y otra con el eje vertical. Además de la cámara, se utilizó una guía de tamaño conocido con la que se obtuvieron los datos necesarios. Dicha guía consiste en un Din A4 de color blanco pegado sobre un cartón para poder mover la guía sin entorpecer la calibración.

Para poder detectar solo el folio blanco en la imagen se tuvo que realizar una nueva segmentación por color, esta vez, bastante fácil al tener que filtrar únicamente por el color blanco, por lo que poniendo la imagen en escala de grises y haciendo una umbralización con los colores más claros se identificaba el rectángulo de la guía. De esta forma se obtenía en color blanco el rectángulo y en color negro todo lo demás.

Pero para evitar que hubiese ruido en la detección por algún color muy claro en el fondo, se volvió a utilizar en esta parte del proceso la lona negra en el fondo, como se puede apreciar en la siguiente imagen.

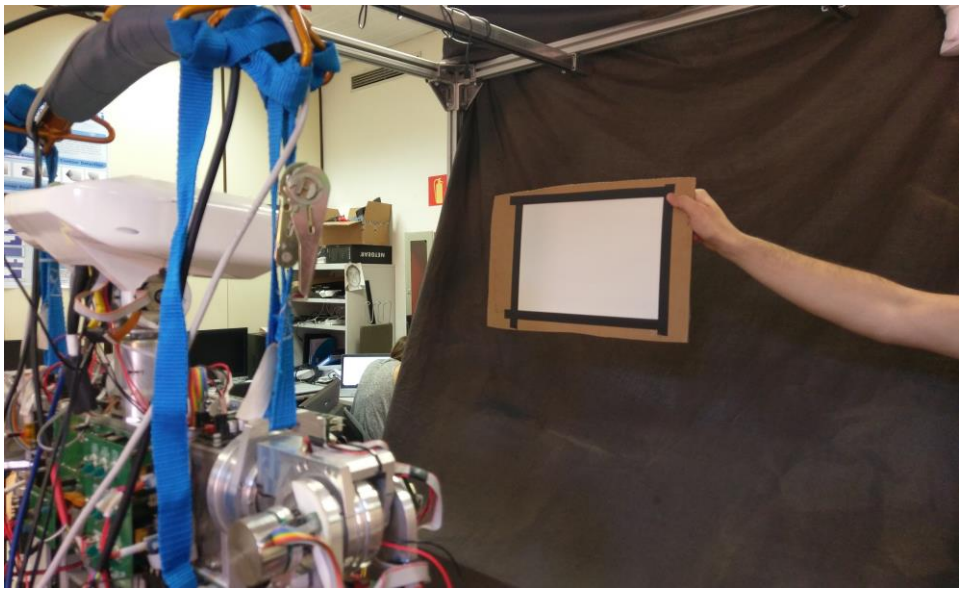

Ilustración 48 – Proceso de calibración con fondo uniforme

Una vez obtenida correctamente la segmentación de la guía, se realizó una función que dibujara un rectángulo que contuviera todo el conjunto de píxeles blancos en la imagen Dicho rectángulo siempre tenía los lados paralelos a los lados correspondientes de la imagen, de esta forma, se obtenía siempre datos verticales y horizontales, sin tener que realizar ninguna transformación.

Con el rectángulo correctamente diferenciado se identificaron los valores, en pixeles, del alto y ancho del rectángulo, necesarios para poder hacer la gráfica.

Finalmente, se calculó el centro del cuadrado para calcular la profundidad a la que se encontraba la guía. Dicho punto se calculaba con la mitad del ancho y alto obtenido anteriormente.

Con los tres datos ya obtenidos, se extrajeron del programa a través de un documento de texto y de esta forma poder calcular la gráfica con una aplicación independiente.

En una hoja de cálculo se introdujeron los datos del ancho y largo del rectángulo obtenido, así como la profundidad a la que se encontraba la guía. Con estos datos se calculó dos rectas: una para el ancho y otra para el largo.

Ambas rectas se obtuvieron con un total de 9 puntos y se pueden apreciar en las siguientes gráficas.

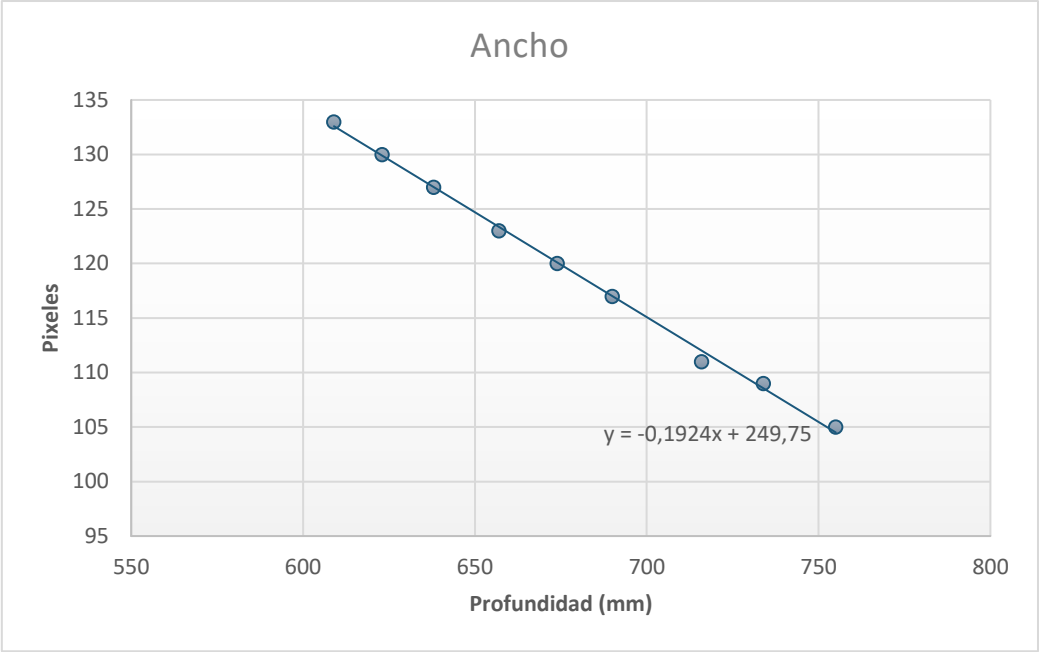

Gráfico 1 – Recta de calibración para el ancho de la guía

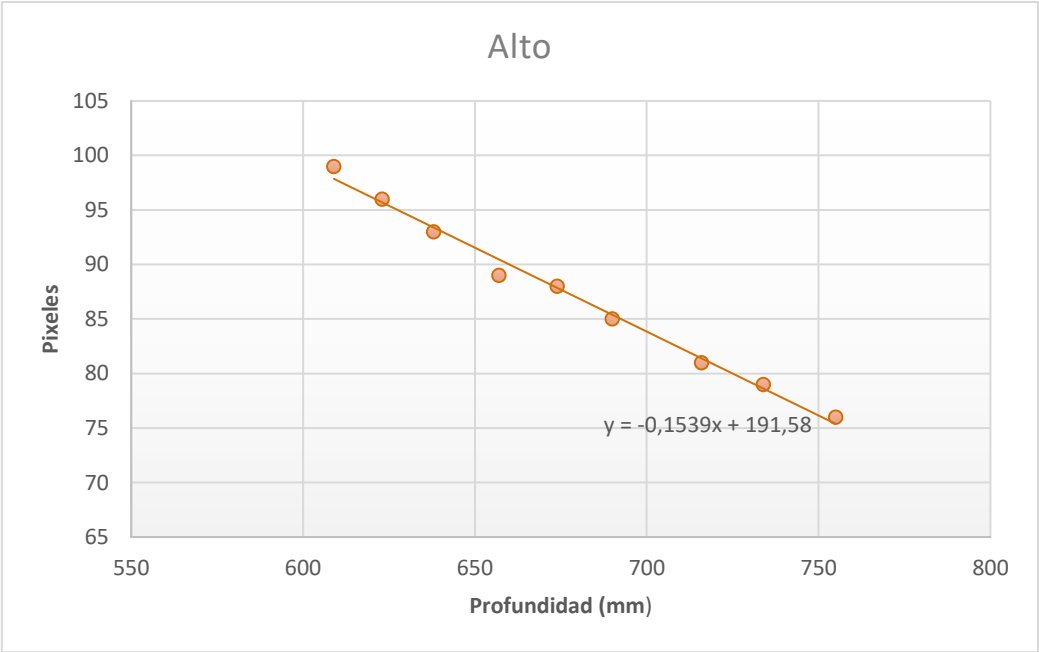

Gráfico 2 – Recta de calibración para el alto de la guía

De las dos rectas se obtuvieron las dos ecuaciones que relacionan los píxeles con la profundidad.

Ya que la guía tiene un tamaño constante, es posible identificar los píxeles que constituirían la guía a una profundidad determinada y, con ello, se podrían calcular relaciones

entre los píxeles y el tamaño de otros objetos, a esa misma profundidad, para finalmente obtener el tamaño real.

4.1.4. Procesamiento 3D

Mediante el procesamiento 3D se realizarán una serie de operaciones y cálculos que, combinadas con la información obtenida en el pre-procesado de la imagen en 2D, permitirán obtener los puntos de interés en el espacio para que el robot pueda coger la caja y manipularla.

A través de una función, se puede obtener la distancia a la que se encuentra un punto de la imagen, siendo ese punto una distancia punto-punto, no siendo en ningún caso paralelo al eje X, Y o Z. Por lo que hace faltas operaciones para poder transformar esta distancia en los valores que necesitamos.

La distancia que se obtiene de la cámara forma círculos, siendo el centro del círculo la cámara, donde todos los puntos del círculo son equidistantes.

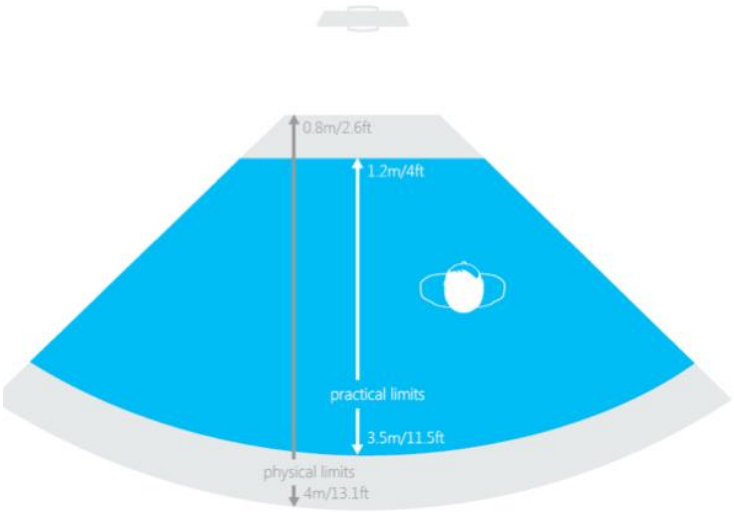

Ilustración 49 – Distancias equidistantes circularmente

Sin embargo, en vez de trabajar con que las distancias sean equidistantes circularmente, se trabajó con distancias equidistantes a un plano. Esto se debe a que trabajar con distancias circulares es muy complicado y acarrearía un coste computacional elevado para la poca mejora en los resultados que se conseguiría. Además, el objeto en la mayoría de los casos estaría situado en el centro de la imagen, siendo esta zona el lugar donde el error es menor.

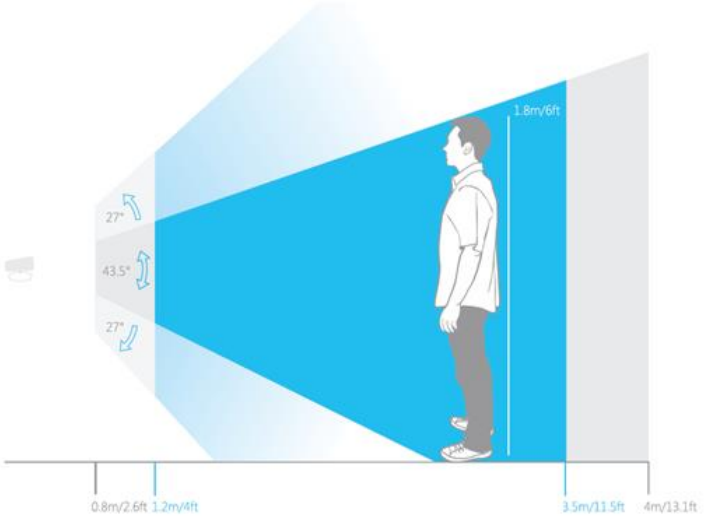

Ilustración 50 – Distancias equidistantes por planos

El primer paso para obtener las distancias es la obtención del punto medio de la imagen, necesaria para utilizar las rectas de calibración. Esta operación se realiza a través de una función en la que se obtiene el ancho y el alto de la imagen para posteriormente dividirlos entre dos.

Debido a que se utilizan formas diferentes para obtener las distancias de los puntos según sea eje X o eje Y y Z, se explicará primero el del eje X y posteriormente el de los otros dos ejes.

4.1.4.1. Obtención eje X

Para obtener el eje X de los puntos solo es necesario utilizar una de las gráficas obtenidas con la calibración. Debido a que el eje X es horizontal al robot, solo se utiliza la gráfica obtenida con el lado horizontal de la guía.

Para explicar el procedimiento de obtención de la distancia real se utilizará la siguiente imagen obtenida experimentalmente, siendo el punto rojo el centro de la imagen y del robot. Por lo que todo lo que esté a la derecha es positivo en el eje X y todo lo que esté a la izquierda es negativo.

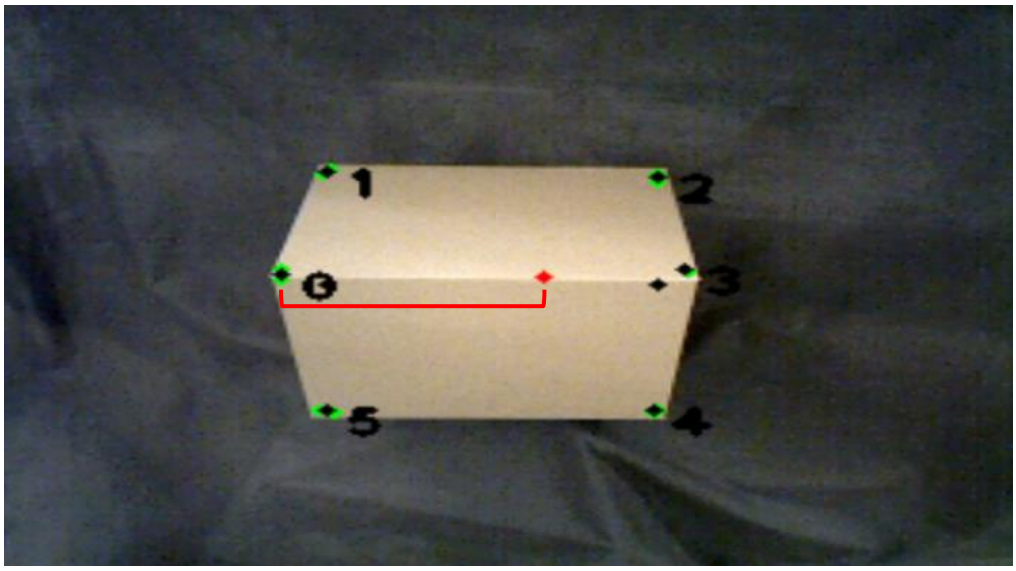

Ilustración 51 – Obtención del eje X

Tomando como ejemplo el punto 0, que es el más situado a la izquierda, la primera acción que se realiza es obtener la profundidad a la que se encuentra dicho punto.

Una vez obtenida esa distancia, se introduciría en la ecuación de la recta obtenida a través de la gráfica 1 de la calibración de la cámara, calculando de esta forma el supuesto valor de los pixeles de la horizontal de la guía (y) a esa profundidad (x).

$$(1) \quad y = x * m + b$$

Donde m es la pendiente de la recta y b es la ordenada en el origen. Sustituyendo los valores se obtiene la siguiente ecuación.

$$(2) \quad pixeles_{guía} = prof * (-0.1924) + 249.75$$

Con esta distancia en píxeles y sabiendo los valores reales de la guía, se obtiene el valor real de la distancia del punto 0 al punto rojo.

Para calcularlo hay que utilizar la fórmula 3 donde $\frac{long\ real_{guia}}{pixeles_{guía}}$ es la relación entre pixeles y mm para todos los puntos a una profundidad x y $pixeles_{cero-centro}$ es la distancia en pixeles entre el punto central y el punto a calcular, siendo siempre la posición del punto real menos la posición del otro punto.

$$(3) \quad long_{cero-centro}(mm) = \frac{long\ real_{guia}}{pixeles_{guía}} * pixeles_{cero-centro}$$

4.1.4.2. Obtención eje Y y eje Z

Para obtener los ejes restantes es necesario utilizar la gráfica que no se ha usado en el apartado anterior, la que se obtiene con el lado vertical de la guía (gráfica 2), y aplicar operaciones trigonométricas. Estos cálculos son necesarios debido a que la distancia obtenida por el sensor de profundidad de la cámara es la distancia perpendicular al plano de la cabeza de TEO (De tal forma que si la cabeza está inclinada un ángulo α respecto a la horizontal, existe ese error), y lo que se quiere obtener es la distancia real desde TEO (coincidente con los ejes de coordenadas).

En este caso, para explicar las obtenciones de los ejes se utilizará el siguiente esquema creado exclusivamente para explicar este apartado:

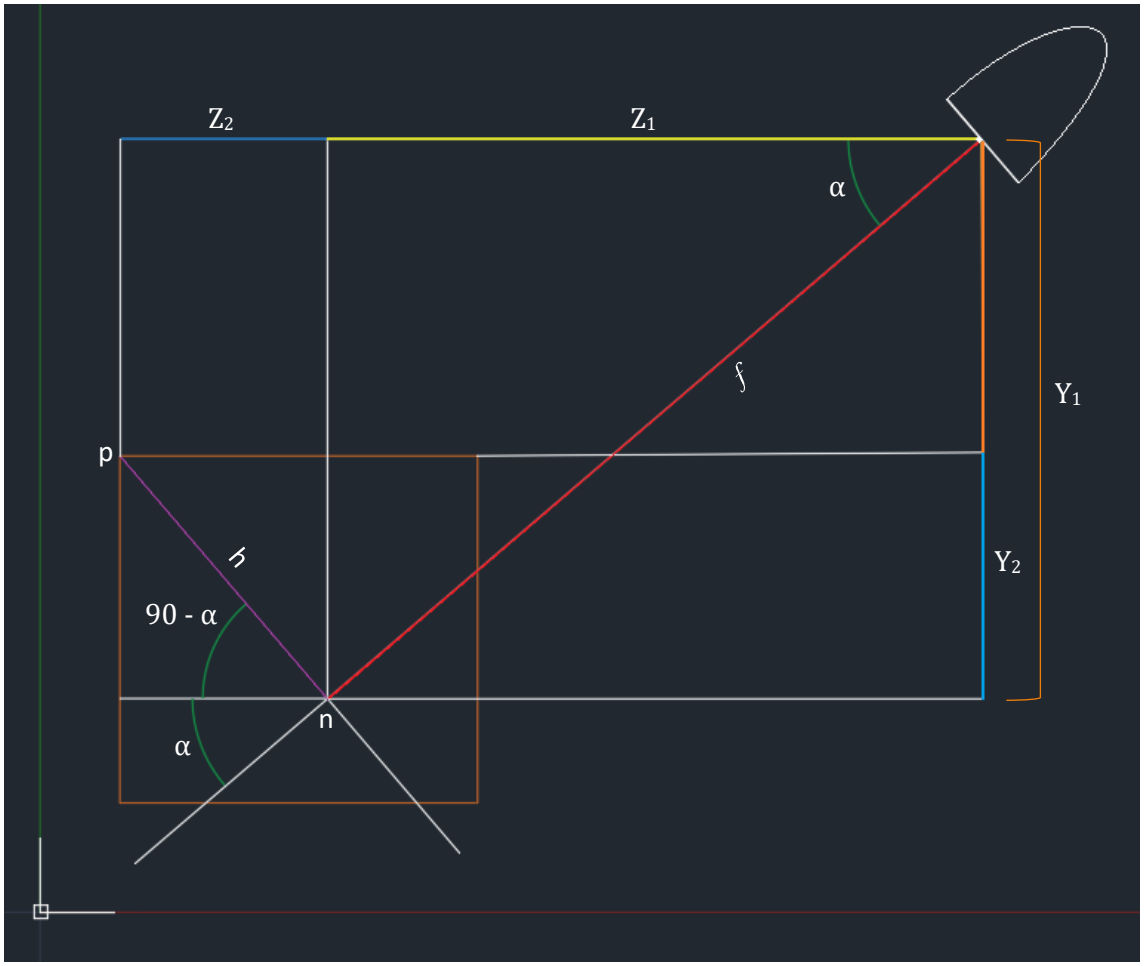

Ilustración 52 – Obtención de los ejes Y y Z

En dicha imagen se puede apreciar la cabeza del robot, situada en la esquina superior derecha, y la caja, un cuadrado de color marrón situado abajo a la izquierda.

Todos los puntos se obtienen de la misma forma. Tanto el eje Y, como el eje Z son la suma de dos valores, uno primero que se obtiene con la profundidad y una función trigonométrica (Z_1 y Y_1), y otro que se obtiene de la profundidad y la gráfica y de una función trigonométrica (Z_2 y Y_2).

$$(4) \quad Z = Z_1 + Z_2$$

$$(5) \quad Y = Y_1 - Y_2$$

Por ello, para explicar el proceso se utilizará como ejemplo el punto denominado p en la imagen.

En un primer lugar se obtiene la distancia al punto p desde la cámara. Esta distancia es igual al punto n debido a que ambos puntos están en el mismo plano perpendicular a la visión del robot. Dicha profundidad es f.

Con la profundidad se calcula de forma directa tanto Z_1 como Y_1 a través de las siguientes ecuaciones trigonométricas, siendo α el ángulo de la cámara respecto al eje de la cámara que es paralelo al suelo:

$$(6) \quad Z_1 = f * \cos(\alpha)$$

$$(7) \quad Y_1 = f * \text{sen}(\alpha)$$

Una vez obtenido Z_1 y Y_1, se calcula Z_2 y Y_2. Para calcular estos valores primero es necesario obtener el valor real de h.

Este valor se calcula de la misma forma que se calcula los valores del eje X, pero con la gráfica de los valores verticales de la guía.

La profundidad, f, se introduciría en la ecuación de la recta obtenida a través de la gráfica 2 de la calibración de la cámara, calculando de esta forma el supuesto valor de los pixeles de la vertical de la guía (y) a esa profundidad (x).

$$(8) \quad y = x * m + b$$

Donde m es la pendiente de la recta y b es la ordenada en el origen. Sustituyendo los valores se obtiene la siguiente ecuación.

$$(9) \quad pixeles_{guía} = f * (-0.1539) + 191.58$$

Con esta distancia en píxeles y sabiendo los valores reales de la guía, se obtiene el valor real de la distancia h.

Para calcularlo hay que utilizar la fórmula 10 donde $\frac{long\ real_{guia}}{pixeles_{guía}}$ es la relación entre pixeles y mm para todos los puntos a una profundidad x y $difference_{pixels}$ es la distancia en pixeles entre el punto central (n) y el punto a calcular (p).

$$(10) \quad h = \frac{long\ real_{guia}}{pixeles_{guía}} * difference_{pixels}$$

En dicha ecuación se debe utilizar la ecuación 11 para calcular el valor de $difference_{pixels}$ debido a los signos en las ecuaciones 4 y 5, ya que si no saldrían valores muy diferentes a los reales.

$$(11) \quad difference_{pixels} = n_y - p_y$$

Una vez obtenido el valor de h correctamente, solo queda volver a realizar una operación trigonométrica en cada eje para obtener los valores de Z_2 y Y_2.

$$(12) \quad Z_2 = h * \cos(90 - \alpha)$$

$$(13) \quad Y_2 = h * \text{sen}(90 - \alpha)$$

Quedando finalmente las ecuaciones 14 y 15:

$$(14) \quad Z = f * \cos(\alpha) + (\frac{long\ real_{guia}}{f * (-0.1539) + 191.58)} * (n_y - p_y) * \cos(90 - \alpha))$$

$$(15) \quad Z = f * \text{sen}(\alpha) + (\frac{long\ real_{guia}}{f * (-0.1539) + 191.58)} * (n_y - p_y) * \text{sen}(90 - \alpha))$$

De esta forma, realizando un bucle para todos los puntos obtenidos previamente, se obtiene la posición real en el espacio de todos los vértices de la caja (pudiendo calcularse también lo que están ocluidos y no han podido ser obtenidos por visión), quedando por calcular únicamente los puntos medios que usara TEO para aproximarse a la caja y posteriormente cogerla.

Sin embargo, debido a que la visión artificial puede variar mucho de un frame a otro, debido a varios factores como las condiciones luminosas o los errores en la cámara, antes de seguir con la obtención de los puntos medios, se realizó en primer lugar una comprobación de los puntos para estar seguro de que los datos obtenidos son coherentes y de esta forma tener mayor fiabilidad a la hora de obtener los resultados finales. En el caso de que no fueran los suficientemente buenos se volvería a realizar todo el proceso llevado a cabo hasta ahora para intentar obtener unos puntos correctos.

4.1.4.3. Comprobación de los puntos

Para asegurar que los resultados de los vértices extraídos de las cajas detectadas tienen unos valores lógicos, se pusieron una serie de restricciones.

Como hay dos formas en las que las cajas pueden estar colocadas, se separaron las restricciones en dos apartados ya que no son comunes todas las restricciones para ambos modos.

Para el modo uno, donde la caja está de lado, se comprueba en un primer lugar que las dos parejas de puntos laterales estén una encima de otra, es decir, que tengan la misma posición o muy parecida en el eje X. Si miramos en la siguiente fotografía los puntos que tienen que tener el mismo valor son los puntos 1 y 2 y los puntos 4 y 5.

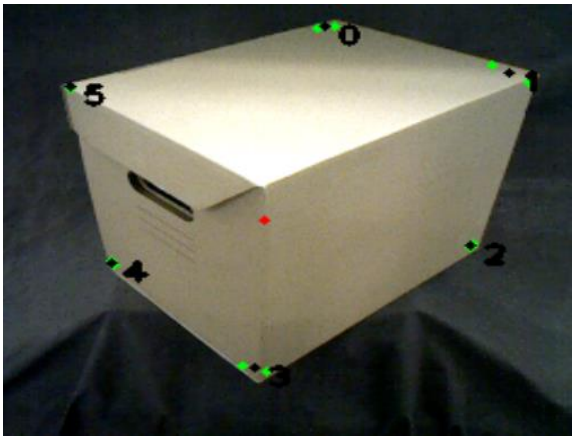

Ilustración 53 – Comprobación de puntos con la caja girada

De esos mismos puntos se comprueban que también tengan la misma profundidad, ya que lo único que cambia en esos puntos es la altura a la que están colocados.

Finalmente, la última comprobación que se realiza en este modo de colocación es que los puntos pertenecientes a los dos planos paralelos al suelo tengan la misma altura o muy parecida. Por lo que los puntos 5, 0 y 1 deberían ser muy parecidos, y los puntos 2, 3 y 4 tendrán que ser también muy parecidos entre ellos, habiendo una distancia considerable entre ambos planos.

Referentes al otro modo, cuando las cajas están de frente, las comprobaciones son parecidas, pero cambian un poco.

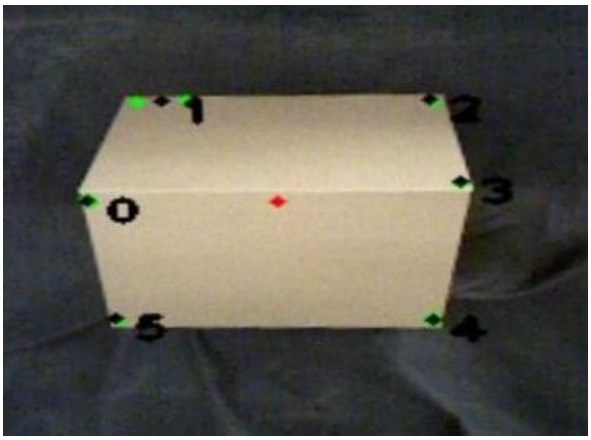

Ilustración 54 – Comprobación de puntos con la caja de frente

Al igual que con el otro modo, la primera comprobaciones son las posiciones del eje X, primero los 3 puntos colocados a la izquierda y luego los 3 puntos colocados a la derecha. Por lo que los puntos 0, 1 y 5 deberían tener la misma posición o parecida, y lo mismo con los puntos 2, 3 y 4.

Una vez comprobado el eje X, se comprueba la altura a través del eje Y. Al igual que con el otro método, se comprueba que los puntos del plano superior y los puntos del plano inferior tengan una altura igual o muy parecida. También se comprueba referente al eje Y que la diferencia de altura entre ambos planos sea superior a 10cm.

Finalmente, solo quedaría probar la profundidad. Para ello se comprueban los cuatro puntos colocados delante, y los 2 puntos colocados en el fondo entre sí. De esta forma se comprueban los puntos los puntos 1 y 2 por un lado, el plano trasero, y los puntos 0, 3, 4 y 5 por otro lado, el plano delantero.

Estas serían todas las comprobaciones realizadas para comprobar que la caja detectada tiene puntos razonables. En el caso de que cualquier condición no se cumpliera, como por ejemplo que un punto de un plano sea muy diferente al resto, se cancelaría los datos obtenidos y se volvería a procesar la caja desde cero. Esto se realizaría hasta que se obtuviera una caja acorde a las condiciones preestablecidas, realizándose posteriormente el cálculo de puntos de aproximado a la caja por el robot.

4.1.4.4. Cálculo de los puntos de aproximación

Una vez calculados los puntos de las cajas y comprobado que dichos puntos obtenidos son coherentes, solo quedan por calcular los datos finales, los puntos que el robot utilizará para acercarse a la caja y poder manipularla.

Los puntos de aproximación que se calculan son todos los puntos medios de las caras laterales de las cajas. No se calculan los puntos medios de las caras superior e inferior debido a que era puntos que el robot no iba a utilizar.

Al igual que con la mayoría de los desarrollos comentados hasta ahora, este proceso también se divide en dos casos de uso según como esté colocada la caja, de frente o girada. Los puntos medios se calculan de uno en uno, utilizando diferentes fórmulas para cada uno.

Para el caso de que la caja estuviera de frente, se calcularía en un primer lugar los puntos medios izquierdo y derecho de la caja, y posteriormente el delantero y posterior, calculando para cada punto su coordenada X, Y y Z por separado.

Antes de empezar a calcular los puntos medios, se definirán una serie de planos sobre la caja que facilitará el trabajo de obtención de dichos puntos. La posición puntos de cada plano no son exactamente iguales, pero sí son muy parecidos gracias a la comprobación, por lo que para obtener dicho plano se calcula la media de los puntos que lo forman. Estos planos solo se pueden utilizar en este modo porque si la caja está girada los planos se tendrían que calcular de otra forma.

El plano superior estaría formado por la coordenada Y de los puntos 0, 1, 2 y 3 mientras que el plano inferior por la coordenada Y los puntos 4 y 5. Adicionalmente, el plano trasero estaría formado por la coordenada Z los puntos 1 y 2 y el plano delantero por la coordenada Z de los puntos 0, 3, 4 y 5. Finalmente, el plano izquierdo está formado por la coordenada X de los puntos 0, 1 y 5 y el plano derecho por la coordenada X de los puntos 2, 3 y 4.

Para el punto más a la izquierda, la coordenada X coincidiría con el plano de la cara izquierda. La coordenada Y sería el plano medio entre el plano superior y el plano inferior. Y, finalmente, la coordenada Z sería el plano medio entre el plano trasero y el plano delantero.

Para el punto más a la derecha el procedimiento sería igual que con el punto medio de la izquierda, pero en la coordenada X en vez de utilizarse el plano de la cara izquierda se utilizaría el plano de la cara derecha.

El punto medio delantero tendría la misma profundidad en el eje Z que el plano delantero, por lo que solo se tienen que calcular los otros dos. La coordenada X equivale al plano medio entre el plano izquierdo y el plano derecho, mientras que la coordenada Y se calcula como el plano medio entre el plano superior e inferior.

De igual manera se calcula el punto medio de la cara trasera de la caja, solo cambia la coordenada Z. En vez de utilizar el plano delantero, se utilizaría el plano trasero, manteniéndose iguales tanto la coordenada X como la coordenada Y.

Con este último punto se han calculado ya los cuatro puntos medios para cuando la caja está de frente. Si la caja está girada la obtención de los puntos medios se realiza de forma diferente, no pudiendo utilizar los planos definidos antes en este caso.

Cada punto medio se calculará uno a uno, en orden, usando como base una esquina de la que se tiene la posición en tres dimensiones y añadiendo diferentes valores para obtener el punto medio.

Primero se obtendrá el punto situado delante a la izquierda, luego el colocado delante a la derecha, para después calcular el que está situado detrás a la derecha y finalmente el de detrás a la izquierda. Este orden se puede apreciar en la siguiente imagen con los puntos rojos.

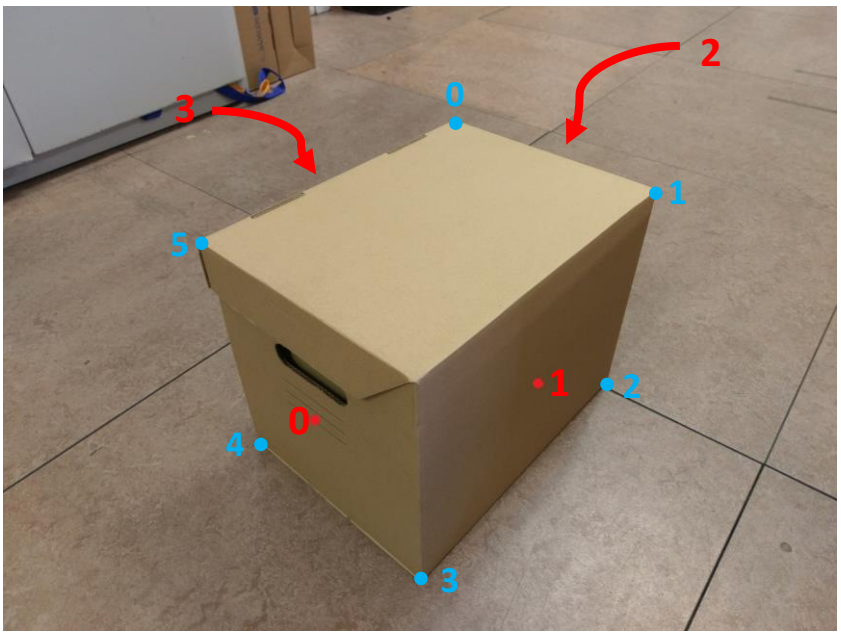

Ilustración 55 – Puntos medios de las caras de la caja cuando está girada

De esta forma, para obtener el punto medio cero, se toma como base el punto tres. A dicho punto se le resta en el eje X la mitad de la diferencia con el punto cuatro, se le resta en el

eje Y la mitad de la diferencia de altura entre el plano superior e inferior y se le suma en el eje Z la mitad de la diferencia entre el punto cuatro y el punto tres.

El resto de los puntos se calculan de la misma manera, se toma como referencia uno ya obtenido y se calcula la mitad de la distancia respecto a otro en los ejes X y Z. Sin embargo, para obtener la posición del punto medio en la coordenada Y se usaría el calculado para el punto medio cero, ya que todos los puntos están situados a la misma altura.

Para el punto medio uno se usa como referencia el punto dos y para obtener los demás valores el punto tres; para el punto medio dos se usa también el dos, pero en este caso en vez del tres como apoyo se usa el cero; y finalmente, para el punto medio tres se usa como referencia el punto cuatro y como apoyo también el punto cero.

Con esto ya se habrían obtenido todos los puntos medios necesario para la manipulación de la caja por parte del robot.

Capítulo 5. Mejoras del proyecto inicial

Con todo el desarrollo anterior ya se cumplieron los objetivos iniciales del proyecto. Sin embargo, para intentar hacer más robusto el algoritmo y obtener buenos resultados en más situaciones se realizó un nuevo filtro para eliminar el fondo de la imagen y quedarnos solo con el objeto a identificar.

Anteriormente ya se habían probado dos filtros diferentes para intentar aislar la caja del entorno, uno por Machine Learning y otro por color, pero con ninguno se obtuvieron unos resultados lo suficientemente buenos como para considerar usarlos. No obstante, en esta ocasión, tras haber experimentado bastante con la cámara RGB-D, se intentó con un filtro de profundidad donde los pixeles que se encuentren a una distancia mayor a un valor preestablecido se pintarán de negro en la imagen.

Antes de intentar este filtro, para aislar la caja se utilizaba una manta negra que hacía el fondo uniforme, como se puede apreciar en la ilustración 39. Pero ahora dicha manta solo se usará en la mesa, dejando el fondo totalmente libre como se muestra en la siguiente imagen.

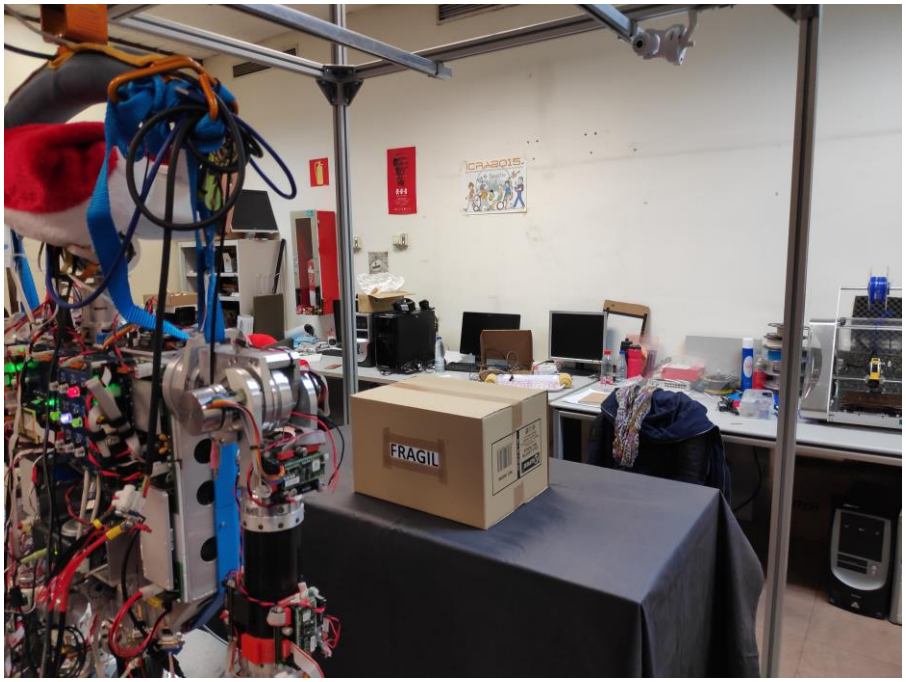

Ilustración 56 – Detección de caja sin fondo uniforme

El filtro de profundidad utiliza el mismo procedimiento que en el apartado de *Procesamiento 3D* cuando se obtiene la posición del eje Z de los vértices de la caja, a diferencia de que no se comprueban solo los seis vértices, sino que se comprueban todos los puntos de la

imagen. Primero se calcula la distancia a la que están todos los objetos de en frente del robot -o el *PointCloud*- para posteriormente, usando la fórmula número 14, obtener la distancia en el eje Z de todos los puntos.

Una vez obtenidos todos los puntos, se tuvo que elegir la distancia a la que se pondría el filtro. La longitud máxima de los brazos de TEO es de 78cm. Sin embargo, para evitar una singularidad en el movimiento del robot (que se produce cuando el robot está en los límites de su espacio de trabajo al perder posibilidades de movimientos) la longitud máxima utilizable de los brazos es algo menor.

De esta forma, teniendo también en cuenta la longitud diagonal de la caja, se eligió una distancia de 1m, asegurando con esa distancia que toda la caja se vería en la imagen y que no se cortaría por estar alguna parte más allá del metro de distancia.

El resultado tras aplicar este filtro se puede apreciar a continuación:

Ilustración 57 – Filtro de Profundidad

Adicionalmente, además del filtro de distancia, se utilizó un algoritmo extra con el fin de obtener la orientación de la caja. Esto es realmente útil ya que se obtendría la información para saber por qué parte habría que abrir la caja.

Se intentaron varios algoritmos entre los que destacan SIFT/SURF y una detección e identificación de texto.

Mientras que con SIFT/SURF se buscaba a tiempo real una imagen previamente seleccionada, con el otro método se buscaba texto en toda la imagen que se analizaba y se comparaba con una base de datos local para comprobar si había alguna coincidencia.

En el caso de los algoritmos SIFT y SURF, se trataba de buscar la imagen de las flechas de frágil que se encuentra en la mayoría de las cajas.

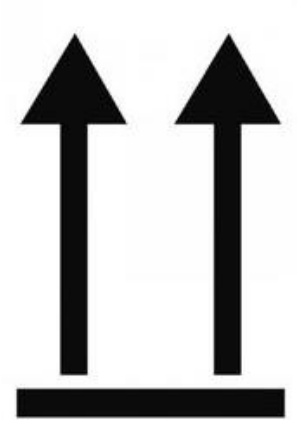

Ilustración 58 – Flechas utilizadas para los algoritmos SIFT/SURF

Sin embargo, tras varias pruebas realizadas se decidió descartar su utilización por la baja detección que se producía con este método debido a la simplicidad de la imagen a buscar. De entre las veces que se obtenía una detección, muchas de ellas eran falsos positivos.

Con el otro algoritmo desarrollado, el de detección e identificación de texto, se consiguieron mejores resultados, utilizando conjuntamente las librerías extras de OpenCV con dos librerías externas de código abierto llamadas Tesseract y Leptonica, que mejoraban la funcionalidad.

El algoritmo solo busca texto que esté colocado de manera horizontal y que no esté dado la vuelta ni girado, por lo que para buscar texto en todas las posiciones posibles se tuvieron que realizar 3 giros a la imagen para comprobar en qué posición está la caja, haciendo un total de 4 búsquedas.

El primer intento se realizaba directamente sobre la imagen que se recibe de la cámara y si no se encuentra ninguna coincidencia, se realizaría un giro de 90 grados para volver a buscar texto. Esto se realizaría durante dos veces más, parándose el proceso en el momento en el que hubiera una correcta detección.

De esta forma, sabiendo en qué giro ha detectado el texto se sabe exactamente la orientación de la caja.

No obstante, el algoritmo no está desarrollado para buscar todo tipos de palabras, solo unas que han sido previamente introducidas en un archivo de texto, entra las que destaca la palabra “FRÁGIL”, siendo la que más aparece en todas las cajas.

Sin embargo, los resultados que se obtenían al principio no eran los esperados, consiguiendo una detección positiva tras muchos intentos. El motivo por el cual se necesitaban

tantos intentos era debido a la baja resolución de la imagen con la que se estaba trabajando hasta ahora.

En todo el desarrollo anterior, para que se pudiera trabajar en tiempo real sin problemas de retraso en el procesamiento de los datos, se decidió que las imágenes de color y de profundidad fueran de un tamaño de 320x240px. Hasta ahora no había surgido ningún problema al utilizar dicha resolución, pero en este apartado, debido a que se tiene que detectar texto, una baja resolución de la imagen implica que el texto sea difícil de identificar, ya que el número de píxeles puede afectar a los datos que se utilizan para comprobar si una detección es una letra o no.

De esta forma, se decidió utilizar otra de las resoluciones disponibles en la cámara ASUS Xtion pro, la de 640x480px (Asus, 2013). Con este nuevo tamaño de imágenes, cuatro veces mayor al anterior, los resultados obtenidos son bastante mejores, detectando el texto normalmente al primer o segundo intento e identificando exactamente la palabra “Frágil”.

Por lo tanto, al identificarse bien el texto, también se identifica correctamente la orientación de la caja.

No obstante, al haberse aumentado en cuatro la resolución de la imagen a color y la de profundidad, el gasto computacional para la obtención y procesado de la imagen ha aumentado considerablemente.

Ilustración 59 – Prueba de detección de texto

Capítulo 6. Pruebas y resultados

En este capítulo se mostrarán todas las pruebas realizadas y los resultados finales obtenidos en cada una de las partes del proceso.

En primer lugar, se enseñarán las pruebas realizadas para las gráficas de calibración, así como el criterio utilizado para la elección de la gráfica utilizada en el proyecto. Posteriormente se analizarán los resultados del procesamiento 3D con la obtención de las distancias y finalmente se mostrarán los resultados de la identificación de la orientación.

6.1. Pruebas de calibración.

La prueba consiste en obtener los valores en pixeles de dos de los lados de una guía a distintas profundidades para después obtener dos gráficas, uno con el lado horizontal y otro con el lado vertical. La guía tiene las dimensiones de un Din A4 cuyos lados son 21cm y 29.7cm.

En un primer momento la guía se coloca perpendicular a la cámara a una distancia pequeña pero dentro del rango de la cámara. Una vez detectada correctamente la guía, se desplazaría de manera constante hacía atrás para capturar más valores. La guía en todo momento está colocada perpendicular a la cámara para que de esta forma asegurar que todos los puntos se encuentran a la misma profundidad.

En total se obtienen tres valores, la distancia en pixeles de los lados verticales y horizontales y la profundidad a la que se encuentra la guía. Todos esos valores son mostrados por pantalla y guardados en un archivo de texto para su posterior utilización en la obtención de las rectas que definen la relación entre píxeles de la guía a una profundidad.

En las siguientes imágenes se puede apreciar el desarrollo de la guía durante una de las pruebas realizadas, siendo la primera imagen con la guía colocada más cerca de la cámara y en la segunda colocada más lejos.

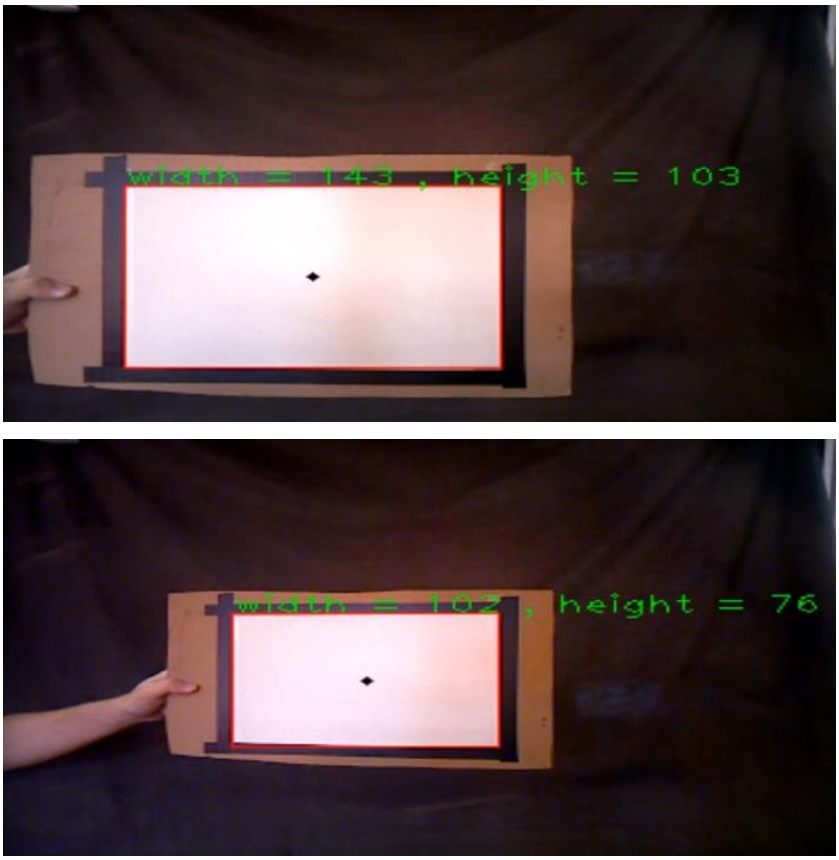

Ilustración 60 – Prueba de calibración de la cámara

Se realizaron un total de cuatro pruebas obteniéndose los valores que se pueden apreciar en la siguiente tabla. Según a la velocidad a la que se moviera hacia atrás la guía se obtenían más o menos valores.

Prueba 1			Prueba 2			Prueba 3			Prueba 4		
Ancho	Alto	Profundidad	Ancho	Alto	Profundidad	Ancho	Alto	Profundidad	Ancho	Alto	Profundidad
133	99	609	134	97	607	136	99	585	142	101	567
130	96	623	130	94	627	132	95	609	131	94	608
127	93	638	127	92	642	128	94	629	125	89	643
123	89	657	121	88	674	124	91	644	121	87	666
120	88	674	117	86	700	122	89	657	117	86	682
117	85	690	112	82	733	120	88	679	113	82	711
111	81	716	109	80	747	117	87	696	107	78	749
109	79	734	107	78	762	114	85	713	106	77	748
105	76	755	104	77	802	112	82	725			
			102	76	813	110	81	734			
						109	80	741			
						107	79	754			
						104	77	769			
						103	76	776			

Tabla 4 – Pruebas de calibración

Y de las cuatro pruebas se calcularon cuatro diferentes gráficas, cada una con las rectas de horizontal y vertical de cada prueba, relacionándose en ellas los pixeles del lado de la guía (eje *y*) con la profundidad a la que se encuentran (eje *x*).

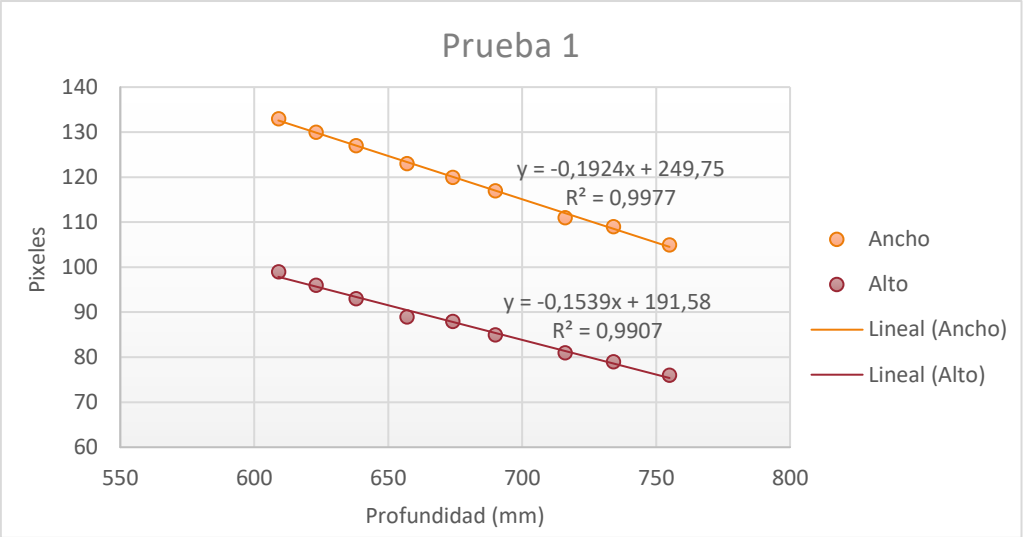

Gráfico 3 – Primera prueba de calibración

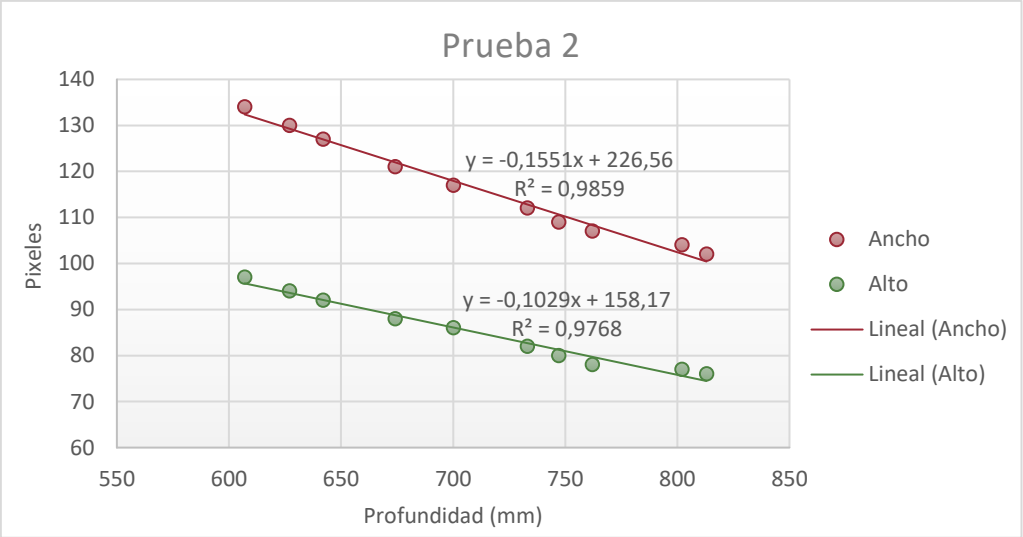

Gráfico 4 – Segunda prueba de calibración

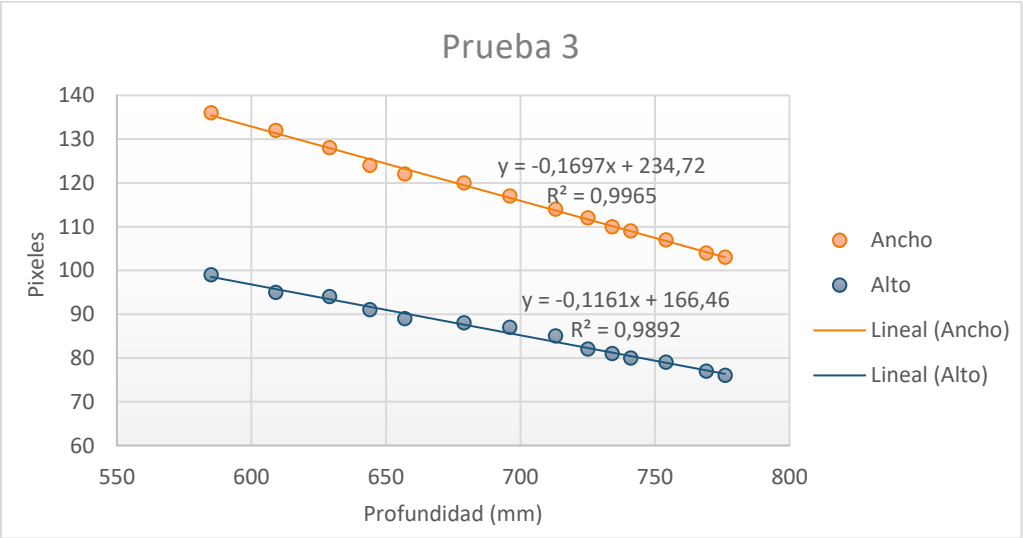

Gráfico 5 – Tercera prueba de calibración

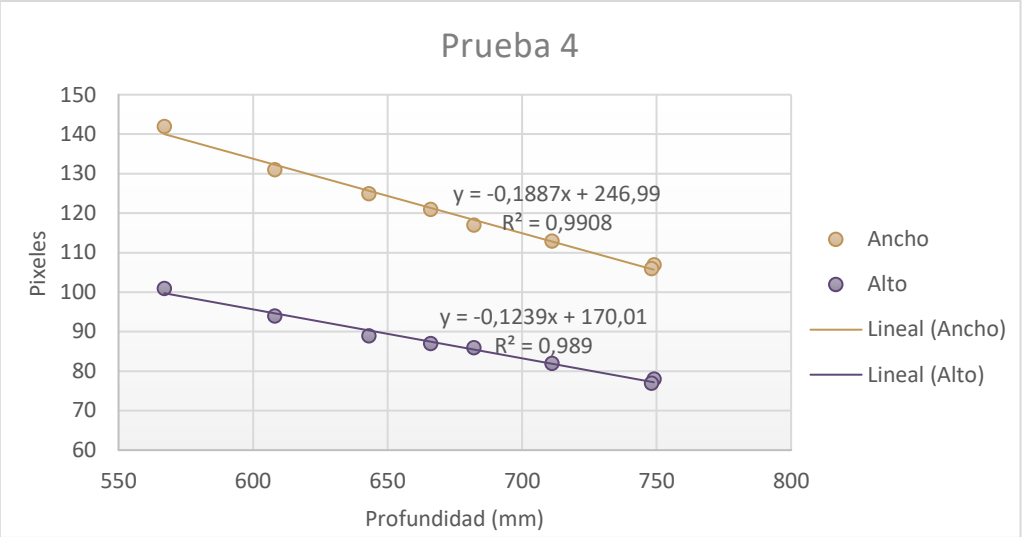

Gráfico 6 – Cuarta prueba de calibración

El criterio utilizado para elegir la recta que se va a utilizar ha sido según el valor del coeficiente de determinación (R cuadrado) el cual determina la calidad de la aproximación lineal. Cuanto más cercano sea el valor a 1 más exacto serán luego los cálculos que se quieran realizar con la fórmula de la recta.

De entre todas las rectas, las que tienen mejor coeficiente de determinación son las de la prueba 1, por lo que dichas rectas son las que se han usado en el resto del desarrollo.

6.2. Pruebas de obtención de las posiciones de la caja en el espacio

En esta prueba se pretendía analizar que los puntos medios de la caja se habían obtenido correctamente. No obstante, en la práctica es muy difícil comprobar que los puntos identificados por el algoritmo desarrollado se encuentran colocados realmente en dicha posición, ya que no existe una forma precisa de obtener su posición real en el espacio.

De forma inicial, se ha empleado un metro durante la obtención de los valores con el fin de comprobar que los puntos medios eran valores lógicos, pudiendo así verificar si el algoritmo funcionaba correctamente. Sin embargo, esto no ha permitido comprobar la ubicación en el espacio con exactitud.

Por ello, para constatar si el software desarrollado obtiene de manera precisa los puntos en el espacio, se ha decidido comprobar en este apartado que las dimensiones de las cajas calculadas a través del algoritmo corresponden con los valores reales.

Durante la prueba se han utilizado un total de tres cajas de diferentes características para tener un mayor número de datos de estudio. Se han realizado 24 comprobaciones con cada caja, 12 con la caja colocada de frente y 12 con la caja girada, realizando un total de 72 comprobaciones en esta prueba.

Las tres cajas que se han utilizado durante esta prueba se pueden apreciar en las siguientes imágenes. Para este apartado se le nombrará a la caja situada arriba a la izquierda como “Caja 1”, a la caja situada arriba a la derecha como “Caja 2” y a la que está más abajo como “Caja 3”.

Ilustración 61 – Cajas utilizadas para la prueba.

La caja 1 tiene unas dimensiones de anchura, longitud y altura de 24.5x33x26cm. Las dimensiones de la caja 2 son de 33.5x42x34cm y de la caja 3 son 30x40x25cm.

En primer lugar, se han analizado los resultados de las tres cajas situadas de frente y posteriormente los obtenidos con las cajas giradas.

Se han obtenido un total de tres gráficos, una para cada caja, donde se comparan de forma porcentual los valores reales de la altura, anchura y profundidad de la caja con el valor máximo, medio y mínimo de los obtenidos a través de la visión artificial.

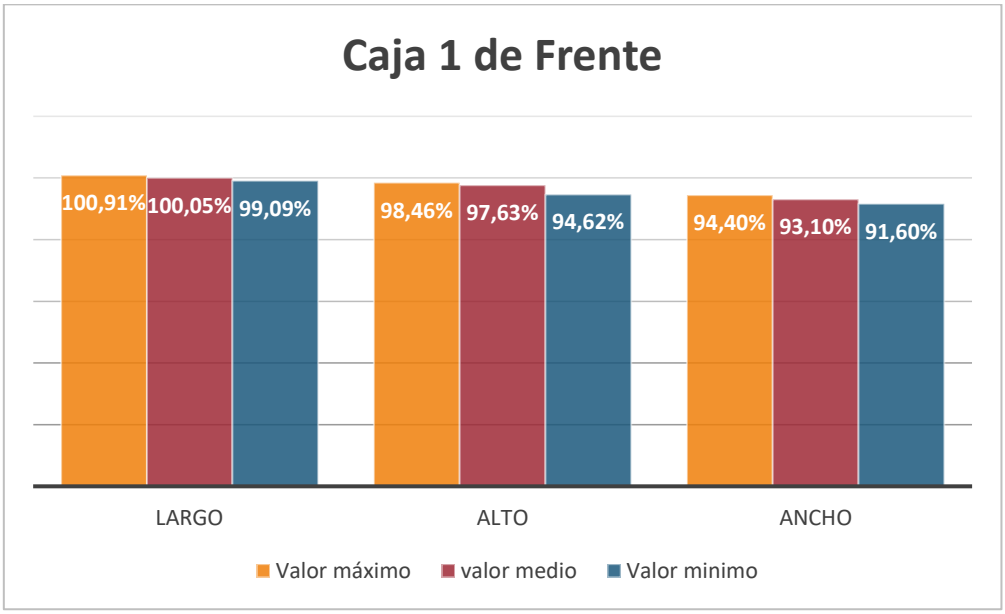

Gráfico 7 – Prueba de obtención de dataos con la caja 1 de frente

En dicho gráfico, con los datos de la caja 1 de frente, se puede apreciar que los valores obtenidos del lado más largo de la caja (el que está de frente a la cámara) son los valores que más se adecuan a la realidad, variando entre 99.09% y 100.91%. Siendo el valor medio casi idéntico al valor real (100.05%).

Referente a la altura, los datos obtenidos también son bastante parecidos a los reales, pero algo peores si los comparamos con los del lado más largo.

El último lado que queda por analizar es el correspondiente a la anchura (viendo la caja de frente corresponde a la profundidad de ésta). Los valores obtenidos son bastante buenos, aunque algo peores que los otros dos. En este caso, los datos varían entre 91.60% y 94.40%, lo que constituye un error de menos de 2cm con respecto al valor real del lado de la caja si se toma como referencia el valor medio.

En términos generales se puede afirmar que los datos obtenidos son bastante parecidos a los reales. El peor dato, como se indicó anteriormente, es el de la profundidad. Sin embargo, son datos muy buenos.

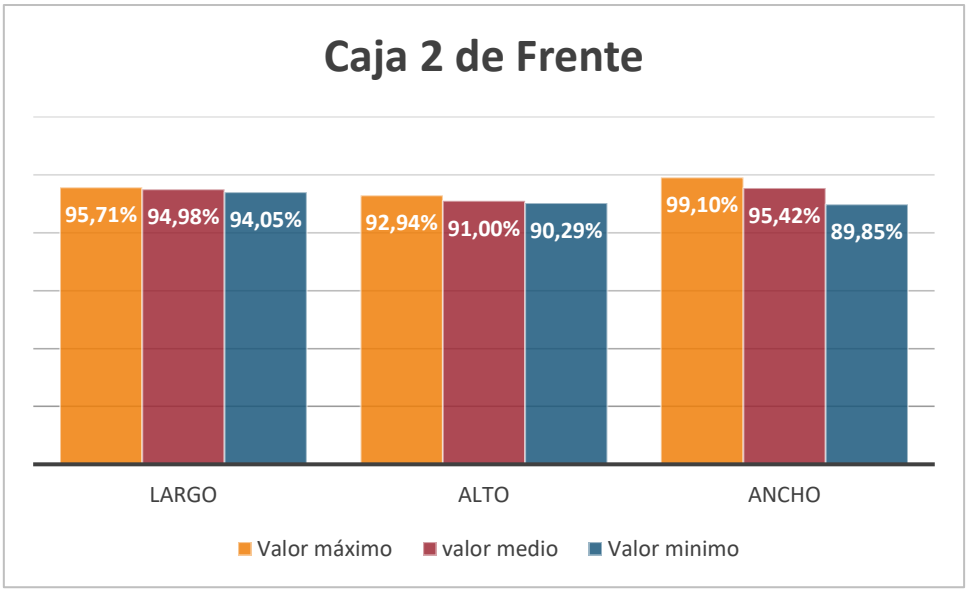

Gráfico 8 – Prueba de obtención de dataos con la caja 2 de frente

En el caso de la caja 2, los datos más parecidos a los reales corresponden a los del ancho de la caja, con un valor medio del 95.42% del valor real. Sin embargo, los datos varían mucho entre ellos, existiendo una diferencia de unos 3cm entre el valor más grande de la prueba y el valor más pequeño.

Referente al lado más largo, el resultado del valor medio es ligeramente inferior que con el lado más ancho (Siendo de 94,88%). No obstante, la variación que existe entre el valor máximo y el mínimo obtenido en los experimentos es la menor de las tres medidas (largo, alto, ancho), variando únicamente entre el 95.71% y 94.05% de sus respectivos valores reales.

Finalmente, respecto a la altura, se han obtenido unos resultados algo peores que con los otros dos lados, con un valor medio del 91%, lo que corresponde a un error de poco más de 3cm con el valor real. En este caso los datos también varían bastante poco.

Los datos de los lados que constituyen la base de la caja, largo y ancho, son resultados muy buenos. En la altura se encuentran los peores resultados, pero aun así no bajan del 90% de similitud, por lo que son resultados que se aproximan bastante a la medida real.

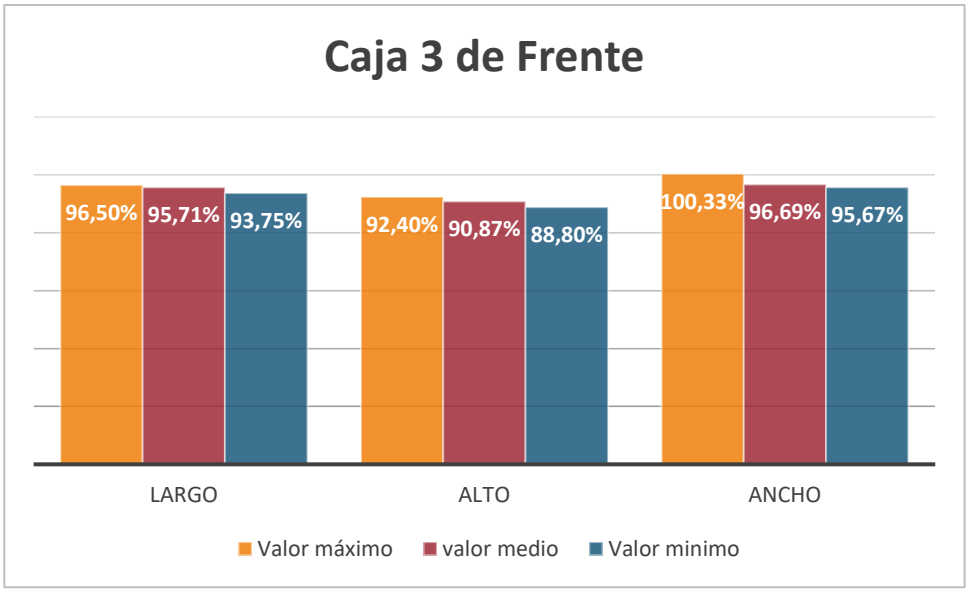

Gráfico 9 – Prueba de obtención de dataos con la caja 3 de frente

En la prueba de la tercera caja, tanto con el lado más largo como con el de la profundidad, se consiguieron resultados bastante buenos superiores al 95% de media en ambos casos.

Siendo algo peores los resultados del lado de la altura, variando los valores entre 88.80% y 92.40%, lo que constituye un error de algo más de 2cm con el valor real si se toma como referencia el valor medio.

Una vez mostrados y verificados los resultados con las cajas de frente, se han realizado las mismas pruebas con las cajas giradas. Sin embargo, en este caso se han analizado las tres cajas a la vez debido a que han cumplido el mismo patrón en ellas.

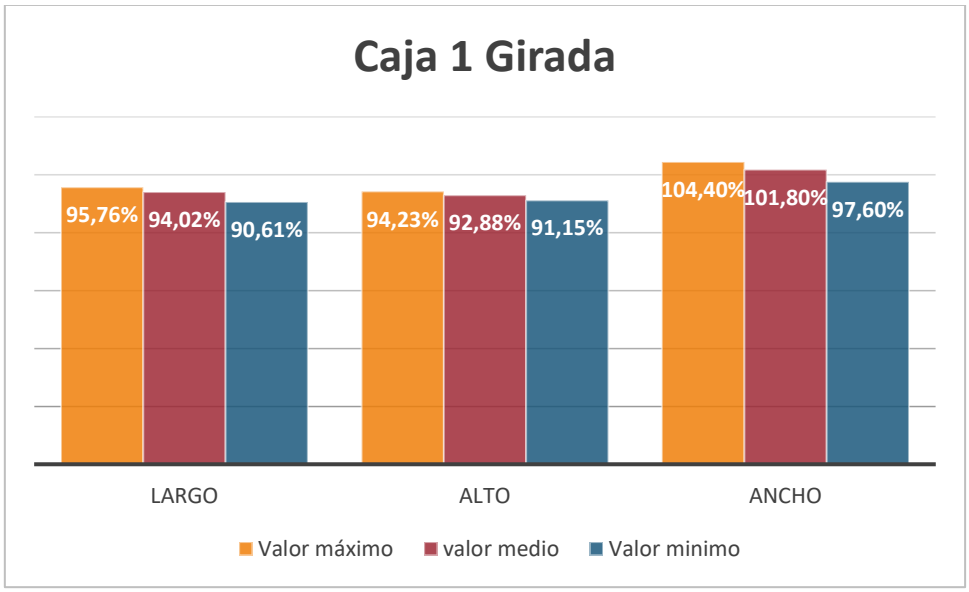

Gráfico 10 – Prueba de obtención de dataos con la caja 1 girada

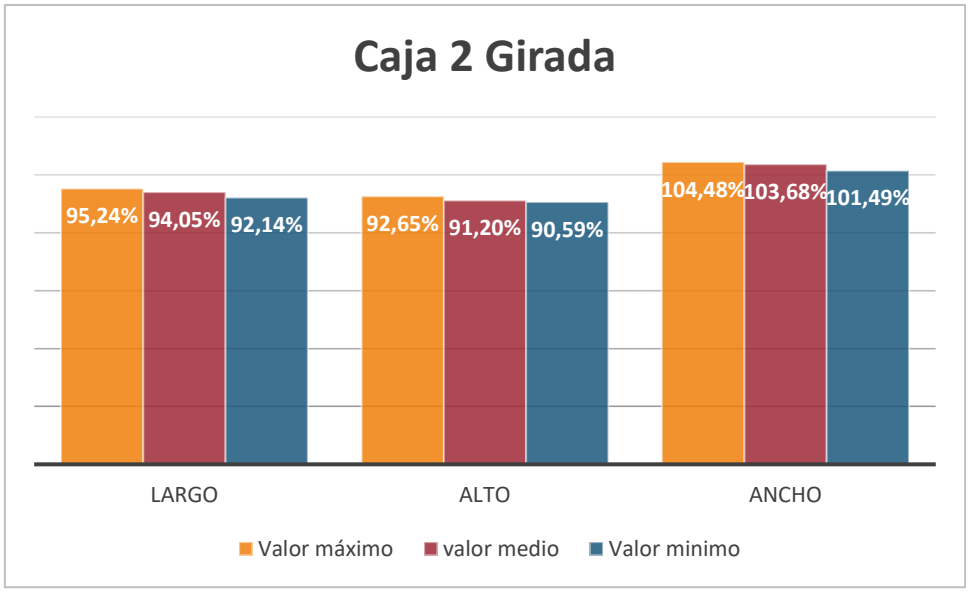

Gráfico 11 – Prueba de obtención de dataos con la caja 2 girada

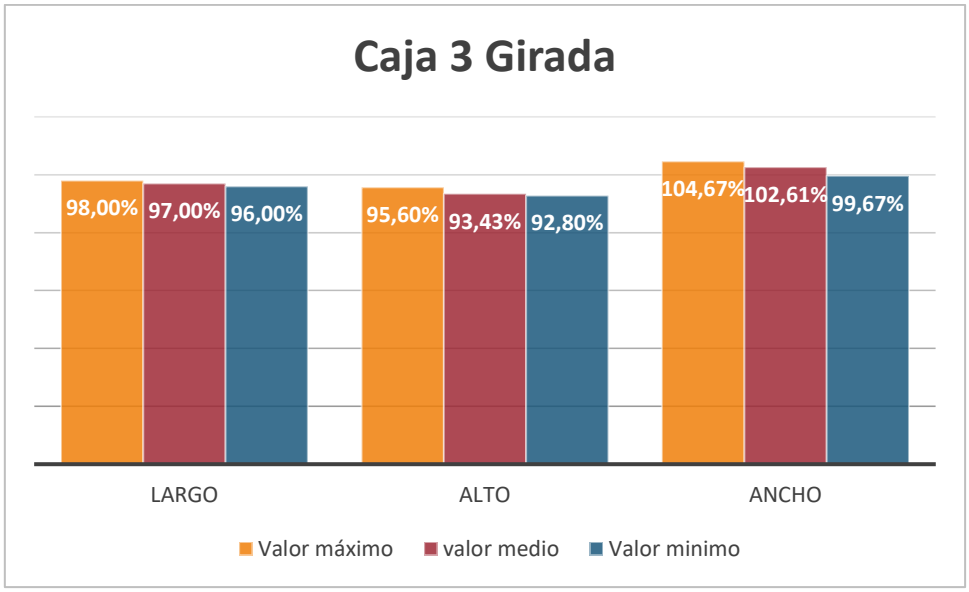

Gráfico 12 – Prueba de obtención de dataos con la caja 3 girada

En las tres pruebas el lado más largo y la altura están siempre por debajo del 100% del valor real y siempre el lado largo es el más parecido al real. Adicionalmente, todos los valores están por encima del 90%, por lo que son valores bastante aproximados.

Respecto al lado que corresponde con el ancho de la caja, en las tres pruebas se obtuvieron valores por encima del valor real. Esto ha podido ser debido a que, para calcular el valor de la anchura de la caja, uno de los lados anchos se queda siempre en la cara de atrás, usando los 3 puntos más alejados de la caja. Estos 3 puntos son los más difíciles de obtener debido al ángulo de incidencia que existe en la cara superior de la caja.

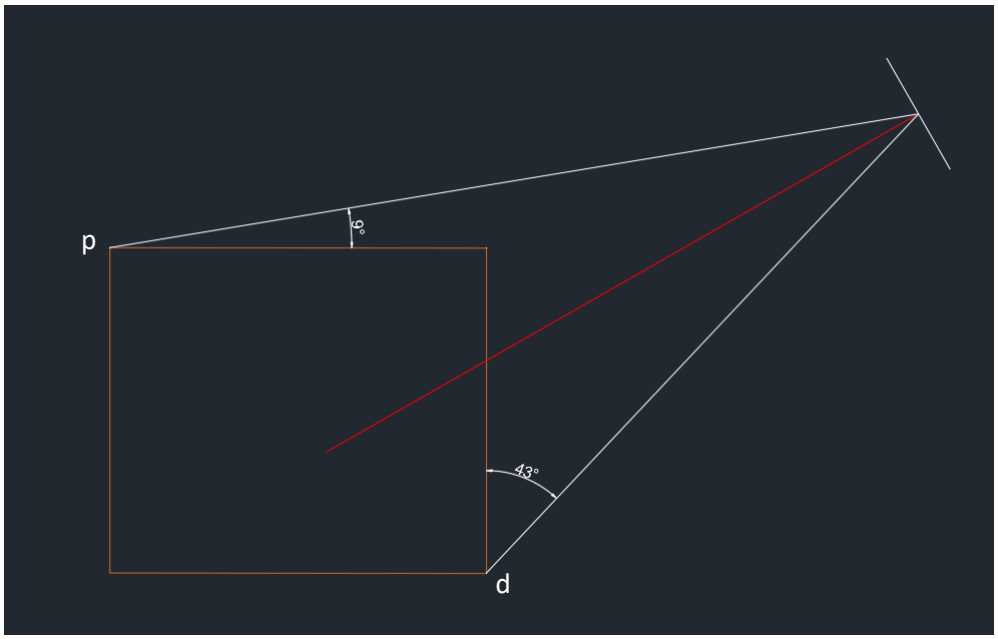

Ilustración 62 – Representación de la diferencia en los ángulos de incidencia

Como ya se dijo en el apartado del *Estado del arte*, la obtención de la profundidad por parte de las cámaras RGB-D se produce con el cálculo del tiempo de vuelo de los rayos infrarrojos, desde que se emiten por la cámara hasta que se reciben por el sensor correspondiente. Dicho retorno se ve retrasado cuanto menor sea el ángulo de incidencia del rayo infrarrojo con el objeto. En la imagen 61 se puede apreciar una representación aproximada de como estaría situada la caja en un caso real. Las dimensiones de la caja no se corresponden con ninguna de las usadas, pero los puntos p y d si están colocados a una distancia en el eje Y y en el eje Z correspondiente a una prueba realizada.

En dicha imagen se puede apreciar la diferencia que existe en el ángulo de incidencia entre la cara de arriba con otra de sus caras. Adicionalmente, se puede afirmar que cuanto más alto o retrasado este el punto p, mayor será su ángulo de incidencia.

Esto último se puede comprobar analizando los datos de la gráfica. El que tiene el mayor error de los tres es la caja 2 (con un valor medio de 103.68%), siendo la caja con la mayor altura. Además, es la caja con los lados más grande por lo que la cara trasera también está colocada más lejos.

Respecto a las otras dos cajas, la caja 3 con un 102.61%, tiene mayor error que la caja 1 con un 101.80%. En este caso la altura es prácticamente igual, pero la caja 3 tiene unas dimensiones mayores por lo que los puntos de la cara trasera están más lejos, y, por lo tanto, tienen un ángulo de incidencia mayor.

Sin embargo, a pesar de estar los valores se encuentran por encima del valor real, son unos resultados bastante parecidos e incluso más cercanos al 100% que los resultados de los otros lados.

De manera global se puede afirmar que, tras las pruebas realizadas, los resultados obtenidos son muy parecidos a los valores reales por lo que el software desarrollado cumple con los objetivos del proyecto.

6.3. Pruebas de orientación de la caja

Finalmente se comprobó el último software desarrollado, el de calcular la orientación de la caja tras identificar texto en ella.

En las siguientes imágenes se puede apreciar varias pruebas realizadas con la caja colocada en diferentes ángulos, identificando en todo momento la palabra *frágil*. Todas las pruebas se realizaron con la mayor resolución posible, la de 640x480px en lugar de 320x240px, para aumentar la detección e identificación de texto.

Ilustración 63 – Pruebas realizadas para obtener la orientación de la caja

Los resultados de detección son bastante buenos, identificando la orientación de la caja sin problemas, pudiéndose usar, teóricamente, sin ningún inconveniente en el proyecto. Sin embargo, debido al aumento del tamaño de la resolución y por lo tanto del tamaño de la imagen que se realizó para poder obtener dichos resultados tan buenos, el tiempo de obtención de la imagen por parte de la cámara es demasiado grande, llegándose a esperar hasta 1 minuto.

Dado que el software se usaría en tiempo real con el robot, una espera tan grande para obtener la información no es útil por lo que esta parte del programa se ha descartado, de momento, ya que no compensa perjudicar la velocidad de todos los programas desarrollados para obtener la orientación de las cajas.

Capítulo 7. Conclusiones y desarrollos futuros

7.1. Conclusiones

Tras la finalización del proyecto se puede concluir que se ha sido capaz de diseñar un algoritmo de visión artificial eficiente que cumpla con los objetivos propuestos al inicio del proyecto: la detección de una caja colocada en el entorno del robot y obtención de sus características geométricas, así como su posición desde al robot.

Al principio del desarrollo se obtuvieron unos resultados que no se esperaban respecto a los algoritmos de Machine Learning, obteniendo muchísimas malas detecciones (falsos positivos) al probarlo. Teóricamente, la mejor solución actual para la identificación de objetos son los basados en capas neuronales, sin embargo, para este proyecto se ha comprobado que la alternativa de operaciones secuenciales funcionaba mejor.

El resto del trabajo ha sido desarrollado sin demasiados inconvenientes, solo es destacable las grandes diferencias que se obtuvieron en los resultados cuando se empezó a trabajar en el laboratorio con la cámara de TEO. Estos peores resultados fueron debidos principalmente a los errores gaussianos que se producían en la cámara y por los pliegues que se encontraban en la lona del fondo, haciendo que se detectasen muchísimos bordes donde no debería con el algoritmo de Canny.

Una vez solucionados estos inconvenientes y finalizado el resto de la implementación (Te lo pongo así para que no se repita desarrollo tan cerca), se puso a prueba el algoritmo desarrollado, comprobando si cumplía los objetivos propuestos.

Aunque no se ha podido verificar de manera precisa que los puntos medios de la caja obtenidos corresponden con los reales, si se pudo comprobar con un metro que dichos puntos eran unos valores lógicos. Adicionalmente, se ha podido comprobar de manera muy satisfactoria que el perímetro de la caja calculado tras el procesamiento 3D, es muy parecido a los valores reales. Asimismo, la obtención de los resultados se consiguió de una manera bastante rápida, aspecto importante a tener en cuenta para un sistema que tiene que trabajar en tiempo real.

Tras todo ello, se puede afirmar que el software realiza correctamente la detección de la caja y la obtención de sus puntos de interés.

Respecto al desarrollo extra realizado hay resultados muy diferentes de un método a otro. Con el filtro de profundidad se consigue eliminar de manera precisa el fondo de la imagen, consiguiendo una correcta segmentación de la caja. Sin embargo, con los dos métodos que se intentaron desarrollar para obtener la orientación, con ninguno de ellos se consiguieron unos resultados lo suficientemente buenos como para poder incluirlo en el programa.

En el segundo de ellos, el de detección e identificación de texto, aunque se obtuvieron unos resultados muy satisfactorios, el tiempo de obtención de la imagen al aumentar su resolución crecía considerablemente, por lo que se tuvo que analizar si merecía la pena ese retraso en todos los desarrollos de visión por aumentar la calidad de la imagen. Finalmente se decidió, que la ganancia en la calidad no era suficiente como para perjudicar al resto de proyectos.

Resumiendo, en términos generales, el software desarrollado realiza de forma satisfactoria los objetivos del proyecto. Sin embargo, se han encontrado varias sorpresas que a priori no se esperaban, destacando el mal funcionamiento experimentado con el algoritmo de Machine Learning, que, sobre papel era la mejor opción, y la gran diferencia que existía entre trabajar con imágenes a trabajar en tiempo real con una cámara como la Asus.

A nivel personal, he encontrado el proyecto bastante gratificante. En primer lugar, me ha permitido aumentar mis conocimientos en la programación y en la visión artificial, sobre todo al trabajar con profundidad y por probar algoritmos basados en capas como Machine Learning que, aunque los resultados no fueran los esperados, me permitieron analizar las dificultades de trabajar con estos métodos. Asimismo, me ha permitido tener el reto de ser parte de un grupo de investigación, así como añadir mi proyecto a una plataforma en continuo desarrollo. Finalmente, este trabajo, con todo lo que implica, me ha dado la oportunidad de comprobar que todo lo que he aprendido tras todas las asignaturas cursadas durante la carrera ha servido para llevarme donde estoy ahora y para prepararme para futuras experiencias en este campo.

7.2. Desarrollos futuros

Como ya se ha mencionado en diferentes partes del trabajo de final de grado, este desarrollo era la primera parte de un proyecto mayor donde TEO, el robot humanoide, tendría que manipular la caja de diferentes maneras, como girarla, abrirla, entre otras, después de la detección de la caja entorno.

Sin embargo, si solo tenemos en cuenta la línea de investigación de este Trabajo de Fin de Grado para los futuros desarrollos, existen varias líneas de trabajos relacionados con la visión artificial que se podrían realizar.

En primer lugar, se tendría que encontrar una forma de identificar de manera más rápida la orientación de la caja. Este desarrollo se podría realizar con una optimización a la hora de obtener las imágenes de mayor resolución de la cámara, disminuyendo el tiempo de espera lo suficiente como para que se pueda usar a tiempo real. Otra posible solución sería desarrollar otro algoritmo que identificara mejor el texto u otros elementos con imágenes de menor tamaño.

Otra línea de trabajo sería modificar el algoritmo de tal forma que pueda detectar todo tipo de cajas sin importar el color de ellas. Actualmente el proyecto detecta todas las cajas en las que su color predominante es el marrón, el blanco o un color claro. Sin embargo, si la caja tiene unas tonalidades muy oscuras, cercanas al negro, es muy probable que no se consiga identificar correctamente la caja.

Finalmente, un último futuro trabajo posible sería desarrollar un algoritmo que pueda identificar diferentes objetos, no solo cajas, diferenciándolos correctamente entre ellos, y calculando de forma precisa los puntos necesarios para su manipulación.

Capítulo 8. Bibliografía

Aguilar, W. G., & Angulo, C. (2012). Compensación y Aprendizaje de Efectos Generados en la Imagen durante el Desplazamiento de un Robot. *X Simposio CEA de Ingeniería de Control*, (págs. 165-170). Barcelona.

Alberich, J., Gomez F., D., & Ferrer F, A. (2011). *Percepción visual.* Barcelona: Universidad Oberta de Cataluña. Recuperado el 18 de Julio de 2018, de https://www.exabyteinformatica.com/uoc/Disseny_grafic/Diseno_grafico/Diseno_grafico_(Modulo_1).pdf

Alexe, B., Desealaers, T., & Ferrari, V. (2010). What is an object? *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (págs. 73-80). San Francisco: IEEE. Recuperado el 26 de Septiembre de 2018, de https://ieeexplore.ieee.org/abstract/document/5540226

Arévalo, V. G. (2004). Arévalo, V., González, J., & Ambrosio, G. (2004). La Librería de Visión Artificial OpenCV. Aplicación a la Docencia e Investigación. *Base Informática, 40*, 61-66. Recuperado el 2 de Diciembre de 2018, de http://mapir.isa.uma.es/varevalo/drafts/arevalo2004lva1.pdf

Asus. (2013). Recuperado el 28 de Septiembre de 2018, de Xtion Pro: https://www.asus.com/es/3D-Sensor/Xtion_PRO/overview/

Ayache, N. (1991). *Artificial vision for mobile robots: stereo vision and multisensory perception.* Londres: Mit Press.

Berriman, R. &. (2017). Will robots steal our jobs? The potential impact of automation on the UK and other major economies. *UK Economic Outlook*, 30-47. Recuperado el Octubre de 2018, de https://www.pwc.co.uk/economic-services/ukeo/pwcukeo-section-4-automation-march-2017-v2.pdf

Bertozzi, M., Broggi, A., & Fascioli, A. (2000). Vision-based intelligent vehicles: State of the art and perspectives. *Robotics and Autonomous systems, 32*(1), 1-16. Recuperado el 11 de Septiembre de 2018, de https://www.sciencedirect.com/science/article/pii/S0921889099001256

Bertozzi, M., Broggi, A., Cellario, M., Fascioli, A., Lombardi, P., & Porta, M. (2002). Artificial vision in road vehicles. *Proceedings of the IEEE, 90*(7), 1258-1271. Recuperado el 11 de Septiembre de 2018, de https://ieeexplore.ieee.org/abstract/document/1032807/

Bloomberg, D. (2018). *Leptonica*. Recuperado el 14 de Noviembre de 2018, de http://www.leptonica.com/

Bradski, G., & Kaehler, A. (2008). *Learning OpenCV.* California: O'Reilly Media, Inc.

Broggi, A., Cerri, P., & Antonello, P. C. (2004). Multi-resolution vehicle detection using artificial vision. *IEEE Intelligent Vehicles Symposium* (págs. 310-314). Parma, Italia: IEEE. Recuperado el 10 de Septiembre de 2018, de https://ieeexplore.ieee.org/abstract/document/1336400/

Buchanan, B. G. (2005). A (Very) Brief History of Artificial Intelligence. *Ai Magazine, 26*(4), 53-60. Recuperado el 25 de Septiembre de 2018, de https://www.aaai.org/ojs/index.php/aimagazine/article/view/1848

Cañas, M. A., Valencia, M., Restrepo, B. J., & Holguín, G. A. (2007). Sistema de visión artificial para el registro de densidad peatonal en tiempo real. *Sciencia et technica, 1*(35). Recuperado el 9 de Septiembre de 2018, de http://revistas.utp.edu.co/index.php/revistaciencia/article/view/5361/2941

Cappella, N. (19 de Enero de 2017). Accenture automates 17,000 jobs without layoffs. *The Stack*. Recuperado el Octubre de 2018, de https://thestack.com/big-data/2017/01/19/accenture-automates-17000-jobs-without-layoffs/

Cho, C. W., Chao, W. H., Lin, S. H., & Chen, Y. Y. (2009). Cho, C. W., Chao, W. H., Lin, S. H., & Chen, Y. Y. (2009). A vision-based analysis system for gait recognition in patients with Parkinson's disease. *Expert Systems with applications, 36*(3), 7033-7039. Recuperado el 9 de Septiembre de 2018, de https://www.sciencedirect.com/science/article/pii/S0957417408006003

Cox, I. J., & Wilfong, G. T. (2012). *Autonomous robot vehicles.* Berlin: Springuer Science.

De la Hescalera, A. (2001). *Visión por computador.* Copia privada con fines docentes: Prentice Hall.

Dobelle, W. H., Quest, D. O., Antunes, J. L., Roberts, T. S., & Girvin, J. P. (1979). Artificial vision for the blind by electrical stimulation of the visual cortex. *Neurosurgery, 5*(4), 521-527. Recuperado el 10 de Septiembre de 2018, de https://academic.oup.com/neurosurgery/article-abstract/5/4/521/2746552

Draghici, S. (1997). A neural network based artificial visión system for licence plate recognision. *Internacional Journal of Neural Systems*, 113-126. Recuperado el 10 de Septiembre de 2018, de https://www.worldscientific.com/doi/abs/10.1142/S0129065797000148

Druzella, U. (2010). Visión Artificial una tecnología industrial. *XVI JORNADES de CONFERÈNCIES JCEE'10. Tendències, aplicacions, recerca i doncencia en el camp de l'Enginyeria Electrònica.* Terrasa: UPB. Recuperado el 12 de Septiembre de 2018, de http://www.crit.upc.edu/JCEE2010/pdf_ponencies/PDFs/09_12_10/Vision%20Artificial%20_UNI%20TERRASSA%202010.pdf

Espasa. (2004). Vista. *Gran Enciclopedia Universal, 18*, 11754-12442. Madrid: Espasa Calpe.

Ferro, G. (2018). Investigación en redes convolucionales y una aplicación a la detección de Peatones. *16th LACCEI International Multi-Conference for Engineering, Education and Technology.* Lima, Peu: IEEE. Recuperado el 24 de Noviembre de 2018, de http://www.laccei.org/LACCEI2018-Lima/student_Papers/SP492.pdf

Gibson, J. J. (1979). *The ecological approach to visual perception.* Boston: Houghton Mifflin.

Gómez, A. M. (2013). *Diseño de robot tipo oruga con brazo articulado*. Tesis Doctoral, Facultad de Mecatrónica, Pereira. Recuperado el 24 de Noviembre de 2018, de https://core.ac.uk/download/pdf/71397865.pdf

González, M. J. (2017). Regulación legal de la robótica y la inteligencia artificial: retos de futuro. *Revista Jurídica de la Universidad de León,*(4), 25-50. Recuperado el Octubre de 2018, de http://revpubli.unileon.es/index.php/juridica/article/view/5285

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* Cambridge: The MIT Press. Recuperado el 26 de Septiembre de 2018, de https://www.synapse.koreamed.org/Synapse/Data/PDFData/1088HIR/hir-22-351.pdf

Hamid, G. (1994). An FPGA-based coprocessor for image processing. *Integrated Imaging Sensors and Processing*, 6/1-6/4. London, Reino Unido: IET. Recuperado el 10 de Septiembre de 2018, de https://ieeexplore.ieee.org/document/383861/

Haralick, R. M., & Shanmugam, K. (1973). Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics,* (6), 610-621.

Herrero, I. (2005). Aspectos de un Sistema de Visión Artificial. *Departamento de Automatización Industrial.* Recuperado el 9 de Septiembre de 2018, de http://www.infoplc.net/files/documentacion/vision_artificial/infoplc_net_Aspectos_de_un_Proyecto_de_Vision_Artificial.pdf

Horn, B. K., & Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence, 17*(1-3), 185-203.

Huerta, H. V., Vásquez, A. C., Dueñas, A. M., Loayza, L. A., & Naupari, P. J. (2009). Reconocimiento de patrones mediante redes neuronales artificiales. *Revista de investigación de Sistemas e Informática, 6*(2), 17-26. Recuperado el 23 de Septiembre de 2018, de http://sisbib.unmsm.edu.pe/bibvirtual/publicaciones/risi/2009_n2/v6n2/a03v6n2.pdf

iPisoft. (13 de Mayo de 2013). Recuperado el 26 de Septiembre de 2018, de Depth Sensors Comparison: http://wiki.ipisoft.com/Depth_Sensors_Comparison

Lam, T. H., Cheung, K. H., & Liu, J. N. (2011). Gait flow image: A silhouette-based gait representation for human identification. *Patter Recognition, 44*(4), 973-987. Recuperado el 7 de Septiembre de 2018, de https://www.sciencedirect.com/science/article/pii/S0031320310004954

Litomisky, K. (2012). Consumer RGB-D Cameras and their. *Rapport Technique, University of California, 20*. Recuperado el 26 de Septiembre de 2018, de https://pdfs.semanticscholar.org/98cf/94d14e97ebf9930bb83d5f989a838ccfee54.pdf

Lucas, B. D., & Kanade, T. (1981). An interative image registration technique with an application in stereo vision. *IJCAI*, 674-679.

Malpartida, E. S. (2003). *SISTEMA DE VISIÓN ARTIFICIAL PARA EL RECONOCIMIENTO Y MANIPULACIÓN DE*. Tesis doctorial, Lima, Peru: PUCP. Recuperado el 12 de Septiembre de 2018, de http://tesis.pucp.edu.pe/repositorio/handle/123456789/68

Mathe, L. S. (2012). Estudio del funcionamiento del sensor Kinect y aplicaciones para bioingeniería. Proyecto final de carrera de Ingeniería en Computación, Universidad Nacional de Córdoba, Córdoba, Veracruz. Recuperado el 15 de Enero de 2019, de https://s3.amazonaws.com/academia.edu.documents/31853932/Especificaciones_Kinect.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1548520054&Signature=drIcMula3P59JiNLVcW74s5vElU%3D&response-content-disposition=inline%3B%20filename%3DEspecificaciones_Kinec

Mendieta, D. (2003). *Reconocimiento de Objetos Bidimensionales en Imágenes mediante la Transformada de Distancia usando Matlab*. Tesis de licenciatura, Universidad de las Américas Puebla, Puebla, Mexico. Recuperado el 16 de Julio de 2018, de http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/mendieta_d_d/

Núñez, P. B.-V. (2011). Robots Sociales para la mejora de la calidad de vida de las personas dependientes. *VI Congreso Iberoamericano de Tecnologías de Apoyo a la Discapacidad IBERDISCAP*, *1*, págs. 94-103. Recuperado el Octubre de 2018, de http://www.academia.edu/download/45316032/Robots_Sociales_para_la_Mejora_de_la_Cal20160503-23243-t0rkxj.pdf

Olmos, R., Tabik, S., & Herrera, F. (2018). Automatic handgun detection alarm in videos using deep learning. *Neurocomputing, 275*, 66-72. Recuperado el 25 de Septiembre de 2018, de https://www.sciencedirect.com/science/article/pii/S0925231217308196

OpenCV. (2018). Recuperado el 2 de Diciembre de 2018, de https://opencv.org/

OpenCV. (2018). *Hough Line Transform*. Recuperado el 18 de Julio de 2018, de https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html

OpenCV. (2018). *OpenCV Contrib*. Recuperado el 2018 de Noviembre de 12, de GitHub Repository: https://github.com/opencv/opencv_contrib

OpenCV Contrib. (2018). Recuperado el 14 de Noviembre de 2018, de GitHub repository: https://github.com/opencv/opencv_contrib

Padilla, C. M. (2016). *Objects Detection using haarscascade algorithms and OpenCV*. Tesis doctoral, Madrid, España: UPM. Recuperado el 13 de Septiembre de 2018, de https://www.researchgate.net/profile/Carlos_M_Padilla/publication/309204287_Objects_Detection_using_haarscascade_algorithms_and_OpenCV/links/58062a6308ae5ad18816240d/Objects-Detection-using-haarscascade-algorithms-and-OpenCV.pdf

Prewitt, J. M. (1970). Object Enhancement and Extraction. *Picture processing and Psychopictorics, 10*(1), 15-19.

Ramírez, J. A., & Chacón, M. I. (2011). Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década. *RIEE & C, revista de Ingeniería Eléctrica, Electrónica y Computación, 9*(1). Recuperado el 22 de Septiembre de 2018, de https://www.researchgate.net/profile/Mario_Chacon/publication/266226481_Redes_neuronales_artificiales_para_el_procesamiento_de_imagenes_una_revision_de_la_ultima_decada/links/565382bb08ae4988a7afacb8/Redes-neuronales-artificiales-para-el-procesamiento-de-

Ratha. (1996). *Computer Vision Algorithms on Reconfigurable Logic Arrays*. Tesis doctoral, Michigan,, USA. Recuperado el 10 de Septiembre de 2018, de https://pdfs.semanticscholar.org/ae45/35cc52e676a9f28d669a2804b872cf854f3e.pdf

Restrepo Arteaga, G. J. (2015). *Aplicación del aprendizaje profundo (deep learning) al procesamiento de señales digitales*. Tesis Universitaria, Universidad Autónoma de Occidente, Santiago de Cali. Recuperado el 25 de Noviembre de 2018, de http://hdl.handle.net/10614/7975

Roberts, L. G. (1963). *Machine perception of three-dimensional solids.* Tesis Doctoral: Massachusetts Institute of Technology.

Roberts, L. G. (1965). Optical and Electro-Optical Information Processing. *Machine Perception of Three-Dimensional Solids*, 159-197. (J. Tippett, Ed.)

Rotman, D. (2013). How Technology Is Destroying Jobs. *Technology Review, 16*(4), 28-35. Recuperado el Octubre de 2018, de http://www.shellpoint.info/InquiringMinds/uploads/Archive/uploads/20130802_How_Technology_is_Destroying_Jobs.pdf

Sanabria S., J. J., & Archila D., J. F. (Diciembre de 2011). Detección y análisis de movimiento usando visión artificial. *Scientia et technica, 3*(49), 180-188. Recuperado el 2018 de Julio de 17, de http://www.redalyc.org/html/849/84922625031/

Sánchez Martín, F. M. (2007). Historia de la robótica: de Arquitas de Tarento al robot Da Vinci (Parte I). *Historia de la robótica: de Arquitas de Tarento al robot Da Vinci (Parte I), 31*(2), 69-76. Recuperado el 24 de Noviembre de 2018, de http://scielo.isciii.es/pdf/aue/v31n2/original1.pdf

Saygin, A. P., Ciceklim, I., & Akman, V. (2000). Turing Test: 50 Years Later. *Minds and Machines, 10*(4), 463-518. Recuperado el 24 de Septiembre de 2018, de https://crl.ucsd.edu/~saygin/papers/MMTT.pdf

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, 61*, 85-117. Recuperado el 25 de Septiembre de 2018, de https://www.sciencedirect.com/science/article/pii/S0893608014002135

Secretaría de Estado de Educación y Formación profesional. (2012). *Vision Artificial: Aplicación práctica de la visión artificial en el control de procesos industriales.* Recuperado el 12 de Septiembre de 2018, de http://visionartificial.fpcat.cat/wp-content/uploads/UD_1_didac_Conceptos_previos.pdf

Shun, R. (2015). *Sistemas de Visión Artificial, Historia, Componentes y Procesamiento de imágenes*. Recuperado el 15 de Julio de 2018, de https://edoc.site/sistemas-de-vision-artificial-historia-componentes-y-procesamiento-de-imagenes-pdf-free.html

Sobel, I. (1970). *Camera Models and Machine Perception.* Tesis Doctoral: Stanford University.

Sucar, L. E., & Gómez, G. (2011). *Visión Computacional.* Puebla, México: Instituto Nacional de Astrofísica, Óptica y Electrónica. Recuperado el 2018 de Julio de 17, de https://ccc.inaoep.mx/~esucar/Libros/vision-sucar-gomez.pdf

Tesseract-OCR. (2018). *Tesseract*. Recuperado el 14 de Noviembre de 2018, de GitHub repository: https://github.com/tesseract-ocr

Ullman, S. (1996). *High-level Vision: Object Recognition and Visual Cognition* (Vol. 2). Cambridge, MA: MIT Press.

Vázquez, E. (2015). *Técnicas de visión artificial robustas en entornos no controlados.* Tesis doctoral, Universidad de Vigo, Vigo, España. Recuperado el 11 de Septiembre de 2018, de https://dialnet.unirioja.es/servlet/dctes?codigo=124604

Villa Moreno, A., Gutiérrez Gutiérrez, E., & Pérez Moreno, J. C. (Junio de 2008). Consideraciones para el análisis de la marcha humana. Técnicas de videogrametría, electromiorafía y dinamometría. *Revista Ingeniería Biomédica, 2*(3), 16-26. Recuperado el 14 de Septiembre de 2018, de http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1909-97622008000100004

Villán, A. F., Acevedo, R. G., Alvarez, E. A., López, A. C., García, D. F., Fernández, R. U., . . . Sánchez, J. M. (2011). Low-cost system for weld tracking based on artificial vision. *IEEE transactions on industry applications, 47*(3), 1159-1167. Recuperado el 11 de Septiembre de 2018, de https://ieeexplore.ieee.org/abstract/document/5727946/

West, D. M. (2015). What happens if robots take the jobs? The impact of emerging technologies on employment and public policy. *Centre for Technology Innovation at Brookings*. Recuperado el Octubre de 2018, de https://www.brookings.edu/wp-content/uploads/2016/06/robotwork.pdf

Wichman, W. M. (1967). *Use of optical feedback in the computer control of an arm.* Stanford University: Departamento de Ingeniería Eléctrica.